7-10-2009

Submodular Percolation

Graham R. Brightwell
London School of Economics

Peter Winkler
Dartmouth College

SIAM J. DISCRETE MATH.
Vol. 23, No. 3, pp. 1149–1178

SUBMODULAR PERCOLATION*

GRAHAM R. BRIGHTWELL† AND PETER WINKLER‡

Abstract. Let $f : \mathcal{L} \to \mathbb{R}$ be a submodular function on a modular lattice $\mathcal{L}$; we show that there is a maximal chain $\mathcal{C}$ in $\mathcal{L}$ on which the sequence of values of f is minimal among all paths from 0 to 1 in the Hasse diagram of $\mathcal{L}$, in a certain well-behaved partial order on sequences of reals. One consequence is that the maximum value of f on $\mathcal{C}$ is minimized over all such paths—i.e., if one can percolate from 0 to 1 on lattice points X with $f(X) \leq b$, then one can do so along a maximal chain. The partial order on real sequences is defined by putting $\langle a(0), a(1), \ldots, a(m)\rangle \preceq \langle b(0), \ldots, b(n)\rangle$ if there is a way to "schedule" the sequences starting at $(a(0), b(0))$ and ending at $(a(m), b(n))$ so that always $a(i) \leq b(j)$. Putting $\mathbf{a} \equiv \mathbf{b}$ if $\mathbf{a} \preceq \mathbf{b} \preceq \mathbf{a}$, each equivalence class has a unique shortest sequence which we call a *worm*. We use the properties of worms to give an efficient method to schedule many real sequences in parallel. The results in the paper are applied in a number of other settings, including obstacle navigation, graph search, coordinate percolation, and finding a lost child in a field.

Key words. submodular function, percolation, modular lattice, graph search

AMS subject classifications. 06C05, 60K35, 90C35, 90B40

DOI. 10.1137/07069078X

1. Introduction.

Submodular systems (distributive lattices on which a submodular "cost" function is defined) arise in many applications (see, e.g., [8, 12, 26]). In some situations it may be desirable to find a path through the lattice which minimizes the maximum cost.

For example, suppose a computer system or network is to be upgraded, and that components must be modified or replaced one at a time without ever allowing the performance of the system to degrade below some fixed threshold. One might imagine that in some circumstance it is necessary to upgrade a component in order to improve its performance, but later downgrade it (allowing it to work better with some not-yet-upgraded component) to prevent system degradation through some critical phase of the conversion.

We show that under the (arguably reasonable) assumption of performance submodularity, no such retreat is ever needed.

More generally, suppose $\mathcal{S}$ is a subset of a finite modular lattice $\mathcal{L}$ with the property that $x, y \in \mathcal{S}$ implies $x \vee y \in \mathcal{S}$ or $x \wedge y \in \mathcal{S}$. Then, if $\mathcal{S}$ contains a path from 0 to 1 in (the Hasse diagram of) $\mathcal{L}$, it contains a maximal chain. (See Figure 1 for a schematic picture of a chain and path in a lattice.)

In most applications of our main theorem the lattice $\mathcal{L}$ is distributive, and in some the cost function is actually modular, that is, $f(x \vee y) + f(x \wedge y) = f(x) + f(y)$. In such a case, $\mathcal{L}$ may be regarded as the lattice of ideals (downward-closed subsets) of a poset X, and the elements of X can be assigned real, not necessarily positive, weights in such a way that $f(x)$ is the sum of the weights of the elements in the ideal x. The

*Received by the editors May 7, 2007; accepted for publication (in revised form) April 3, 2009; published electronically July 10, 2009.

http://www.siam.org/journals/sidma/23-3/69078.html

†Department of Mathematics, London School of Economics and Political Science, WC2A 2AE London, UK (G.R.Brightwell@lse.ac.uk).

‡Department of Mathematics, Dartmouth College, Hanover, NH 03755-3551 (peter.winkler@dartmouth.edu). This author's research was supported by NSF grant DMS-0600876 and in part by a Clay Senior Fellowship at the Mathematical Sciences Research Institute, Berkeley, CA.

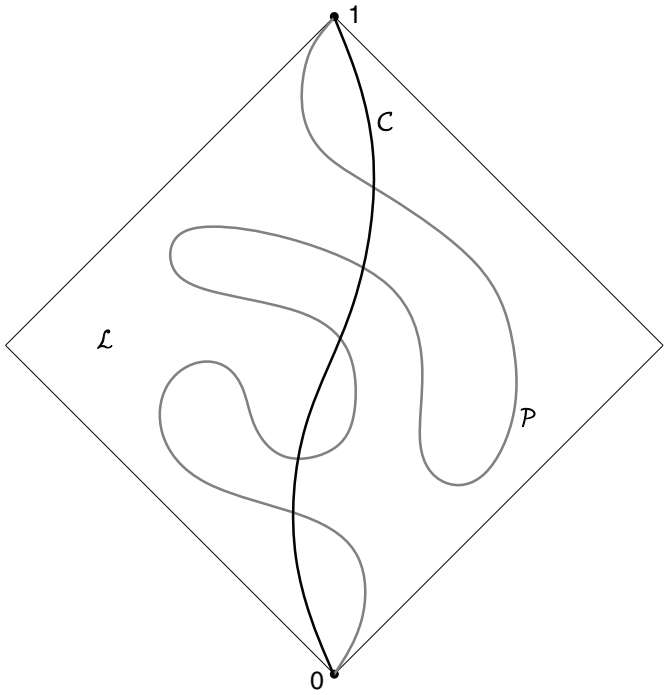

FIG. 1. *There is a chain $\mathcal{C}$ which beats any path $\mathcal{P}$ in the "worm order."*

conclusion here is that there is a linear extension of X, the weights of whose initial segments are no more than the maximum weight of the ideals in any path from 0 to 1 in $\mathcal{L}$.

As we shall see, the theorem guarantees the existence of a maximal chain in any submodular system which not only minimizes the maximum value of f over all 0-1 paths, but the minimum value as well, and in fact minimizes the whole sequence of values of f in a useful partial order which we call the "worm order." One consequence is that a multiplicity of real sequences can be scheduled—in a manner to be made precise below—so as to minimize their maximum sum, in time polynomial in the sum of their lengths.

We think of one word $\mathbf{a}$ (a finite sequence of real numbers) as being "below" another word $\mathbf{b}$, and write $\mathbf{a} \preceq \mathbf{b}$, if there is a way of scheduling the two sequences so that, at any time, the entry of $\mathbf{a}$ is below the entry of $\mathbf{b}$. The relation $\preceq$ is reflexive and transitive; we consider two words $\mathbf{a}$ and $\mathbf{b}$ to be *equivalent* if $\mathbf{a} \preceq \mathbf{b}$ and $\mathbf{b} \preceq \mathbf{a}$. The set of equivalence classes thus forms a partial order.

We make these notions precise in sections 2 and 3, and we show that each equivalence class contains a unique shortest word, called a "worm." In particular, all words in an equivalence class have the same maximum value as the worm representing the class. We then obtain a natural notion of the addition of two or more worms, which is associative. We give details in section 4. Our contention is that this is a natural arithmetic structure on equivalence classes of sequence.

In particular, a question that arises naturally in many contexts is that of finding

a word of a certain type whose maximum value is minimized. As we shall show, it is often no harder, and more useful, to find a minimum word in the worm order.

In section 5, we consider submodular functions on modular lattices. In a modular lattice, all maximal chains have the same length, and we can consider the word obtained by looking at the function value along a chain. We show that there is a chain whose word is below or equal to that of all other chains, in the worm order. In section 6, we further show that this word is below that of all *paths* from 0 to 1 in the Hasse diagram of the lattice. A related result had earlier been proved by Bienstock and Seymour [3], as we describe in that section.

In section 7, we show that, nevertheless, the problem of finding a chain whose maximum f-value is minimized is NP-hard, even in the case where f is a modular function on a distributive lattice.

In section 8, we turn to applications. One of our main motivations is to study problems set in the plane, where our "lattice elements" are east-west paths between two fixed points s and t, and a "chain" moves the path upwards from south to north. As a specific instance, if we have an acyclic planar directed network, with s as source and t as sink, then the set of s-t paths naturally forms a distributive lattice, and the length of the path is a modular function, so our results apply. We consider other problems of a similar type, as well as problems of searching a graph, and problems in percolation.

2. Scheduling words. A (real) *word* is a finite nonempty sequence of real numbers. We use $\mathbb{R}^*$ to denote the set of all words. Our words will be written as, e.g., $\mathbf{a} = \langle a(0), \dots, a(m)\rangle$, and we shall occasionally refer to $a(\ell)$ as the ℓ-entry of the word $\mathbf{a}$. We define the *length* of $\mathbf{a} = \langle a(0), \dots, a(m)\rangle$ to be m (not $m+1$) and denote $\max_{j=0,\dots,m} a(j)$ simply by $\max \mathbf{a}$.

Let $[n]$ be shorthand for the set $\{0, 1, 2, \dots, n\}$; we denote by $\mathcal{L}(m, n)$ the set of (oriented) lattice paths which proceed north and east from $(0, 0)$ to (m, n) in the grid $[m] \times [n]$.

If $\mathbf{a} = \langle a(0), \dots, a(m)\rangle$ and $\mathbf{b} = \langle b(0), \dots, b(n)\rangle$ are elements of $\mathbb{R}^*$ and

$$\varphi = \langle (x_0, y_0), (x_1, y_1), \dots, (x_{m+n}, y_{m+n})\rangle \in \mathcal{L}(m, n)$$

is a lattice path, then we may define a sum

$$\mathbf{s} = \mathbf{a} +^{\varphi} \mathbf{b}$$

by $s(k) := a(x_k) + b(y_k)$, which we may think of as the result of summing $\mathbf{a}$ and $\mathbf{b}$ pairwise using φ as a "schedule." Each time k increases by 1, one of x_k and y_k increases in φ, and the schedule then calls for an advance in the corresponding word $\mathbf{a}$ or $\mathbf{b}$, while the other word stays at its current position.

We also write $\varphi(k) = (x_k, y_k)$ for $k = 0, \dots, m+n$, and we will occasionally refer to the sequences $\varphi_x = \langle x_0, x_1, \dots, x_{m+n}\rangle$ and $\varphi_y = \langle y_0, y_1, \dots, y_{m+n}\rangle$.

The schedule and the addition can conveniently be represented in tabular form, as in the example below. We can represent the sum $\mathbf{s} = \mathbf{a} +^{\varphi} \mathbf{b}$, where $\mathbf{a} = \langle 0, 2, 1\rangle$, $\mathbf{b} = \langle 0, -1, 3, 0\rangle$, and $\varphi = \langle (0,0), (0,1), (1,1), (2,1), (2,2), (2,3)\rangle$, by means of the table on the left below, or (more concisely, and with no real loss of information) by the table on the right.

k	0	1	2	3	4	5
$a(x_k)$	0	0	2	1	1	1
$b(y_k)$	0	−1	−1	−1	3	0
$s(k)$	0	−1	1	0	4	1

$\mathbf{a}$	0	0	2	1	1	1
$\mathbf{b}$	0	−1	−1	−1	3	0
$\mathbf{a} +^{\varphi} \mathbf{b}$	0	−1	1	0	4	1

If we regard $\mathbf{a}$ as a process in time which uses some resource at rate $a(i)$ during stage i, then $\mathbf{a}+^{\varphi}\mathbf{b}$ uses the resource at rate $s(k)$ during the kth stage of the combined processes. It may be desirable to choose the schedule φ so that $\max \mathbf{s}$ is minimized. (In the previous example, the schedule φ is not optimal; there is an alternative schedule where this maximum is just 3.) Although there are $\binom{m+n}{m}$ possible schedules, it is not surprising that this optimization problem is easy to solve, for example by dynamic programming. However, bounded look-ahead is not good enough; one might even need to see all of both words before scheduling the first advance.

What is perhaps surprising is that there is a way to choose φ which not only minimizes $\max \mathbf{s}$, but also allows the sum $\mathbf{s}$ to be used in place of $\mathbf{a}$ and $\mathbf{b}$ in minimizing the maximum sum of $\mathbf{a}$, $\mathbf{b}$ and any number of additional summands. One consequence is that optimal scheduling of q words of lengths m can be done in time polynomial in q and m (assuming that there is a bound on the space needed to represent each input real).

We now make this notion of optimality precise. Suppose that $\mathbf{a}_1, \ldots, \mathbf{a}_q$ are words of lengths $m_1, \ldots, m_q$, respectively. Let $\psi \in \mathcal{L}(m_1, \ldots, m_q)$ be a lattice path from $(0, \ldots, 0)$ to $(m_1, \ldots, m_q)$ on $[m_1] \times \cdots \times [m_q]$, with

$$\psi = \langle (x_0^1, \ldots, x_0^q), (x_1^1, \ldots, x_1^q), \ldots, (x_M^1, \ldots, x_M^q) \rangle,$$

where $M = \sum_{i=1}^{q} m_i$, and define

$$\sum_{i \in \{1,\ldots,q\}}^{\psi} \mathbf{a}_i$$

to be the word $\mathbf{s}$ of length M given by

$$s(k) = \sum_{i \in \{1,\ldots,q\}} a_i(x_k^i)$$

for $k = 0, \ldots, M$.

In the terminology of percolation, the schedule ψ is an oriented path in $\mathbb{Z}^q$ and we may reasonably ask whether there can be an *unoriented* path from $(0, \ldots, 0)$ to $(m_1, \ldots, m_q)$ on $[m_1] \times \cdots \times [m_q]$ with a smaller maximum than any oriented one. In fact, this is not the case. Moreover, as we will see, this result applies in much more general situations as well.

There is another liberalization of the notion of lattice path which we can make without affecting the minimax, and that is to allow diagonal steps (in other words, to allow the schedule ψ to advance in any number, or even in all, of the words simultaneously). This is easy to see: such a diagonal step can be split into orthogonal steps by advancing next, at each stage, in the word not yet advanced whose net change in value is lowest.

Returning to our theme of scheduling to minimize the maximum, we will prove the following result.

THEOREM 2.1. *For any words $\mathbf{a}_1$, $\mathbf{a}_2 \in \mathbb{R}^*$ of lengths m_1 and m_2, there is a schedule $\varphi \in \mathcal{L}(m_1, m_2)$ with the following property. For any $q > 2$, and any words $\mathbf{a}_3$, $\mathbf{a}_4, \ldots, \mathbf{a}_q$ (of lengths $m_3, \ldots, m_q$), we have*

$$\min_{\psi \in \mathcal{L}(m_1,\ldots,m_q)} \max \sum_{i \in \{1,\ldots,q\}}^{\psi} \mathbf{a}_i = \min_{\chi \in \mathcal{L}(m_1+m_2,m_3,\ldots,m_q)} \max \left((\mathbf{a}_1 +^{\varphi} \mathbf{a}_2) +^{\chi} \sum_{i \in \{3,\ldots,q\}}^{\chi} \mathbf{a}_i \right).$$

Furthermore, the schedule φ can be obtained in polynomial time.

In other words, the schedule φ is uniformly optimal, over all sums including both $\mathbf{a}_1$ and $\mathbf{a}_2$.

3. The calculus of worms. Let us denote by $\mathbf{0}$ *any* word all of whose entries are 0. For a word $\mathbf{a} = \langle a(0), \dots, a(m)\rangle$, we define the word $-\mathbf{a} = \langle -a(0), \dots, -a(m)\rangle$. Observe that, for words $\mathbf{a}$ and $\mathbf{b}$, a schedule φ for which each entry of $\mathbf{a} +^{\varphi} (-\mathbf{b})$ is nonpositive is exactly one for which $a(\varphi_x(k)) \le b(\varphi_y(k))$ for each k. It is thus natural for us to write "$\mathbf{a} \preceq \mathbf{b}$" if such a φ exists; the symbol "$\preceq$" then defines a preorder (that is, a transitive, reflexive binary relation) on $\mathbb{R}^*$. More precisely we have the following lemma.

LEMMA 3.1. *Let* $\mathbf{a}$, $\mathbf{b}$, *and* $\mathbf{c}$ *be real words of arbitrary lengths. Then we have the following:*

1. $\mathbf{a} \preceq \mathbf{a}$*;*
2. $\mathbf{a} \preceq \mathbf{0}$ *if and only if* $a(i) \le 0$ *for each* $i = 0, \dots, m$*;*
3. $\mathbf{a} \preceq \mathbf{b}$ *if and only if there is a schedule* φ *such that* $\mathbf{a} +^{\varphi} (-\mathbf{b}) \preceq \mathbf{0}$*;*
4. $\mathbf{a} \preceq \mathbf{b} \preceq \mathbf{c}$ *implies* $\mathbf{a} \preceq \mathbf{c}$*;*
5. $\mathbf{a} \preceq \mathbf{b}$ *and* $\mathbf{c} \preceq \mathbf{d}$ *implies* $\mathbf{a} \mid \mathbf{c} \preceq \mathbf{b} \mid \mathbf{d}$*, where "* $\mid$ *" denotes concatenation;*
6. $\mathbf{a} \preceq \mathbf{b}$, $\mathbf{b} +^{\varphi} \mathbf{c} = \mathbf{d}$ *implies that there is a schedule* ψ *for which* $\mathbf{a} +^{\psi} \mathbf{c} \preceq \mathbf{d}$.

Proof. For the first statement, we put $\varphi(2i) = (i, i)$, $\varphi(2i+1) = (i, i+1)$ if $a(i) \le a(i+1)$, and $\varphi(2i+1) = (i+1, i)$ otherwise. It then follows that $a(\varphi_x(k)) \le a(\varphi_y(k))$ for each k.

The second statement is evident since all schedules are equivalent when one of two summands is a constant; the third was observed above. For transitivity, let φ and ψ realize $\mathbf{a} \preceq \mathbf{b}$ and $\mathbf{b} \preceq \mathbf{c}$, respectively, and now schedule all three words together in such a way that $\mathbf{b}$ advances only when both φ and ψ call for it. The resulting schedule, restricted to just $\mathbf{a}$ and $\mathbf{c}$, proves the desired inequality.

The table below illustrates this argument in the case where $\mathbf{a} = \mathbf{c} = \langle 1, 5, 0, 2\rangle$, and $\mathbf{b} = \langle 1, 4, 5, 2, 3, 0, 2\rangle$.

$\mathbf{a}$	1	1	1	1	5	0	0	0	0	0	0	0	2
$\mathbf{b}$	1	1	4	5	5	5	2	3	0	0	0	2	2
$\mathbf{c}$	1	5	5	5	5	5	5	5	5	0	2	2	2

The last two statements are obvious and easily checked. □

It is evident that if $\mathbf{a} \preceq \mathbf{b}$, where $\mathbf{a}$ has length m and $\mathbf{b}$ has length n, then $a(0) \le b(0)$ and $a(m) \le b(n)$; also, $\max \mathbf{a} \le \max \mathbf{b}$ and $\min \mathbf{a} \le \min \mathbf{b}$. These conditions are not sufficient, however; e.g., the words $\langle 0, 1, -1, 0\rangle$ and $\langle 0, -1, 1, 0\rangle$ are not comparable. On the other hand, as shown in the example above, $\langle 1, 5, 0, 2\rangle$ and $\langle 1, 4, 5, 2, 3, 0, 2\rangle$ are comparable in *both* directions, and we shall think of these as equivalent.

Accordingly, define $\mathbf{a} \equiv \mathbf{b}$ if $\mathbf{a} \preceq \mathbf{b}$ and $\mathbf{b} \preceq \mathbf{a}$, and let $\mathbb{W} := \mathbb{R}^*/\equiv$ be the set of equivalence classes. On $\mathbb{W}$, the relation $\preceq$ becomes a genuine partial order, where $x \preceq y \preceq x \implies x = y$. We shall see that there is a natural normal form for $\mathbb{W}$, that is, a (rather squiggly) canonical representative for each equivalence class.

Let $\mathbf{a}$ be a word of length m. A subinterval $[j, k]$ (in $\mathbb{Z}$) of the index interval $[0, m]$ will be called a *spanner* if $k - j > 1$ and $a(i) \in [\min(a(j), a(k)), \max(a(j), a(k))]$ for each $i \in [j, k]$.

Suppose $\mathbf{a}$ contains a spanner $[j, k]$, and let $\mathbf{a}'$ be defined by $a'(i) = a(i)$ for $i \le j$ and $a'(i) = a(k+i-j-1)$ otherwise. (Thus $\mathbf{a}'$ is the result of removing the interior of the spanner.) Then we claim $\mathbf{a}' \equiv \mathbf{a}$. To see that $\mathbf{a} \preceq \mathbf{a}'$, consider a schedule in which each entry of $\mathbf{a}$ is scheduled against the corresponding identical entry of $\mathbf{a}'$, except

that all of $a(j+1), \ldots, a(k-1)$ are scheduled against the larger of $a'(j) = a(j)$ and $a'(j+1) = a(k)$. Similarly, to see that $\mathbf{a}' \preceq \mathbf{a}$, we do the same thing except that we schedule $a(j+1), \ldots, a(k)$ against the smaller of $a'(j) = a(j)$ and $a'(j+1) = a(k)$.

The example mentioned in the proof of Lemma 3.1 also serves as an illustration of this argument. In the word $\mathbf{b} = \langle 1, 4, 5, 2, 3, 0, 2\rangle$, the subintervals $[0, 2]$ and $[2, 5]$, corresponding to the sections $\langle 1, 4, 5\rangle$ and $\langle 5, 2, 3, 0\rangle$, respectively, of $\mathbf{b}$, are spanners: removing the interiors of both spanners yields the worm $\mathbf{a} = \langle 1, 5, 0, 2\rangle$, and the table illustrates the proof that $\mathbf{a} \equiv \mathbf{b}$.

THEOREM 3.2. *Each equivalence class in* $\mathbb{W}$ *contains a unique shortest word, which we call a* worm. *A word* $\mathbf{w}$ *is a worm (for some class) if and only if*

1. $w(i+1) - w(i)$ *is never zero and alternates in sign; and*
2. *up to (and including) some entry* $w(\ell)$*, each entry of* w *is either higher than all previous entries or lower than all previous entries; at* $w(\ell)$ *and thereafter, each entry is higher than all* following *entries or lower than all following entries.*

An entry of a word $\mathbf{w}$ that is either higher than all previous entries, or higher than all subsequent entries, will be called a *peak*. An entry that is either lower than all previous entries, or lower than all subsequent entries, is a *valley*. Exceptionally, the initial entry in a worm will only be called a peak if it is followed by a lower entry, and a valley if it is followed by a higher entry, with a similar convention for the final entry of a worm.

Typically there are two (consecutive) indices which can serve as the "ℓ" of the theorem. For either of them, the part of $\mathbf{w}$ up to $w(\ell)$ is called the "expanding phase" of $\mathbf{w}$, the rest the "contracting phase." A typical worm looks something like the one whose graph is illustrated in Figure 2 below.

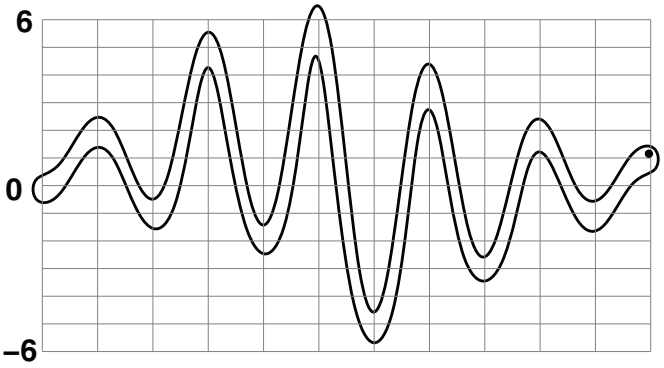

FIG. 2. *A "graph" of the worm* $\langle 0, 2, -1, 5, -2, 6, -5, 4, -3, 2, -1, 1\rangle$.

There are two nontrivial cases where the index "ℓ" of the theorem is unique. One is where there are two indices which are tied for the maximum, e.g., $\langle 0, 1, -1, 1, 0\rangle$, in which case only $\ell = 2$, the location of the unique minimum, will serve. The other is the symmetric situation where there are two indices tied for the minimum. These possibilities will never cause us any real difficulties, although they will occasionally require a little special attention in our proofs.

Proof. We have already seen that if $\mathbf{a}$ contains a spanner, it cannot be a worm,

since removing the interior of the spanner yields a shorter word in the same class. Note that a repeated entry ($a(i+1) = a(i)$) lies in a spanner, unless the worm has only two entries, in which case we can collapse them to one by hand. Thus, a worm must indeed satisfy the first condition of the theorem.

To get the second, let $w(k)$ be the first entry of $\mathbf{w}$ which does not set a record, that is, which is not greater or less than all previous entries. We may assume by symmetry that $w(k) > w(k-1)$; therefore, it must be that $w(k) \leq w(k-2)$. In that case $w(k+1) > w(k-1)$ else $[k-2, k+1]$ would be a spanner; but we then apply a similar argument one index higher to get $w(k+2) < w(k)$, and continuing in this fashion shows that we are now and forever in the contracting phase. It follows that the second condition of the theorem holds, with $\ell = k-1$.

Since eviscerating spanners (and collapsing double entries) must eventually result in a word satisfying the conditions of the theorem, it suffices for the uniqueness part of the theorem to show that two distinct such words cannot be equivalent.

Suppose that $\mathbf{a} \neq \mathbf{b}$, and both satisfy the conditions of the theorem. We may assume $\mathbf{a}$ has length m and $\mathbf{b}$ length n, with $m \geq n$. We may also assume $a(0) = b(0)$, $a(m) = b(n)$, $\max \mathbf{a} = \max \mathbf{b}$, and $\min \mathbf{a} = \min \mathbf{b}$, else equivalence is immediately ruled out.

Let ℓ be the first index for which $a(\ell) = \max \mathbf{a}$, and suppose that for some $i < \ell$, $a(i) \neq b(i)$; let j be the smallest such i. Suppose first that $a(j) > b(j)$, yet $\mathbf{a} \preceq \mathbf{b}$ via some schedule φ.

We claim that $\varphi(2i) = (i, i)$ for each $i = 0, \ldots, j-1$. We prove this by induction on i: it is certainly true for $i = 0$. If the claim is true for some value $i < j-1$, then $\varphi(2i+1)$ is equal to either $(i, i+1)$ or $(i+1, i)$. If $\varphi(2i+1) = (i, i+1)$, then we have $b(i+1) = a(i+1) > b(i) = a(i)$, but then $b(i+2) \leq a(i+2) < a(i)$, since $\mathbf{a}$ is still expanding. Hence we cannot have $\varphi(2i+2) = (i, i+2)$, and we must have $\varphi(2i+2) = (i+1, i+1)$. Similarly, if $\varphi(2i+1) = (i+1, i)$, then $b(i+1) = a(i+1) < b(i) = a(i)$, so again since $\mathbf{a}$ is expanding, $a(i+2) > a(i) = b(i)$. Hence we cannot have $\varphi(2i+2) = (i+2, i)$, and so in this case we also have $\varphi(2i+2) = (i+1, i+1)$. The claim now follows by induction.

It follows that $\varphi(2j-2) = (j-1, j-1)$. If $a(j) > a(j-1)$, we must have $\varphi(2j-1) = (j-1, j)$ and $\varphi(2j) = (j-1, j+1)$, implying that $b(j+1) \geq a(j-1) = b(j-1)$ and thus $\mathbf{b}$ is now contracting—but then $\mathbf{b}$ can never attain $\max \mathbf{a}$. If, on the other hand, $a(j) < a(j-1)$, we must have $\varphi(2j-1) = (j, j-1)$ and $\varphi(2j) = (j+1, j-1)$, implying that $a(j+1) \leq b(j-1) = a(j-1)$ which directly contradicts the expansion of $\mathbf{a}$.

A dual argument shows that, again assuming $j < \ell$, when $a(j) < b(j)$ we cannot have $\mathbf{b} \preceq \mathbf{a}$.

What if the first i for which $a(i) \neq b(i)$ is $i = \ell$? Then $\mathbf{b}$ has not reached its maximum yet; hence, switching the roles of $\mathbf{a}$ and $\mathbf{b}$, we again have $i < \ell$.

If there is an i with $a(i) \neq b(i)$ which is in the *contracting* phase of $\mathbf{a}$, we reverse the time-orders of both $\mathbf{a}$ and $\mathbf{b}$ to return to the case $i < \ell$.

The only remaining possibility is that the two worms are identical from their left (stage zero) ends up to their max, and from their right ends as well. Since $\mathbf{a} \neq \mathbf{b}$, at least one of them (say, $\mathbf{a}$) hits its max twice. Its value between those two stages is the unique $\min \mathbf{a}$; since $\min \mathbf{b} = \min \mathbf{a}$, the worm $\mathbf{b}$ must behave identically and we have arrived at a contradiction which proves the theorem. □

We have now proved that every word $\mathbf{a}$ is in the equivalence class of exactly one worm. We call two words *worm-equivalent* if they are in the same equivalence class.

Lemma 3.1 and Theorem 3.2 tell us something, but not everything, about how

to compare two worms. If, reading from left to right in their expanding phases, the first difference occurs at a point where $a(i) > b(i)$, we can conclude $\mathbf{a} \not\preceq \mathbf{b}$; similarly reading from right to left in their contracting phase. However, this does not mean that the existence of *some* j for which $a(j) > b(j)$ prevents $\mathbf{a} \preceq \mathbf{b}$, even if the worms are of the same length. For example, take $\mathbf{a} = \langle 0, -2, 2, -3, 3\rangle$ and $\mathbf{b} = \langle 0, -1, 1, -2, 3\rangle$. Then $a(2) > b(2)$, but it is easily checked that $\mathbf{a} \preceq \mathbf{b}$.

We give the following result, characterizing exactly when two worms are comparable.

THEOREM 3.3. *Let* $\mathbf{a} = \langle a(0), \ldots, a(m)\rangle$ *and* $\mathbf{b} = \langle b(0), \ldots, b(n)\rangle$ *be two worms. Then* $\mathbf{b} \preceq \mathbf{a}$ *if and only if none of the following holds:*

(i) $\max \mathbf{b} > \max \mathbf{a}$*;*
(ii) $\min \mathbf{b} > \min \mathbf{a}$*;*
(iii) *there is a peak* $b(j)$ *of* $\mathbf{b}$*, and a valley* $a(i)$ *of* $\mathbf{a}$*, both in expanding phase, with* $b(j) > a(i-1)$ *and* $a(i) < b(j-1)$*;*
(iv) *there is a peak* $b(j)$ *of* $\mathbf{b}$*, and a valley* $a(i)$ *of* $\mathbf{a}$*, both in contracting phase, with* $b(j) > a(i+1)$ *and* $a(i) < b(j+1)$*.*

Here, a peak of $\mathbf{b}$ is in expanding phase if it lies before the first global minimum of $\mathbf{b}$, and in contracting phase if it lies after the last global minimum. Similarly, a valley of $\mathbf{a}$ is in expanding phase if it comes before the first global maximum of $\mathbf{a}$, and in contracting phase if it lies after the last global maximum. In (iii), if i and/or j is 0, then $a(i-1)$ is to be interpreted as equal to $a(i) = a(0)$, and/or $b(j-1)$ as $b(j) = b(0)$, and similarly for condition (iv).

Proof. Let us first show that, if any of the four conditions holds, then $\mathbf{b} \not\preceq \mathbf{a}$. For the first two conditions, this is obvious.

Suppose that $b(j)$ and $a(i)$ are as in (iii). Then the peak $b(j)$ is higher than any $a(i-1)$, and so is higher than any of the peaks preceding $a(i)$ in $\mathbf{a}$. Thus, in any schedule φ witnessing $\mathbf{b} \preceq \mathbf{a}$, $b(j)$ must be scheduled strictly later than $a(i)$. Similarly, the valley $a(i)$ is lower than $b(j-1)$, and so φ must schedule $a(i)$ strictly later than $b(j)$. Therefore no such schedule φ exists.

A symmetrical argument shows that (iv) also implies that $\mathbf{b} \not\preceq \mathbf{a}$.

Now suppose that none of (i)–(iv) holds. We construct a schedule φ witnessing $\mathbf{b} \preceq \mathbf{a}$.

For every peak $b(j)$ in the expanding phase of $\mathbf{b}$, we schedule $b(j)$ against the first possible peak of $\mathbf{a}$, i.e., the peak $a(k)$ such that $a(k) \geq b(j) > a(k-2)$. Condition (i) assures us that every peak in the expanding phase is scheduled. Suppose we schedule the peak $b(j)$ against $a(k)$, and the lower peak $b(j-2)$ against $a(k')$, where necessarily $k' \leq k$. Then we claim that the valley $b(j-1)$ lies below all entries of $\mathbf{a}$ between $a(k')$ and $a(k)$ inclusive. If not, then $k' < k$, and the lowest point on this segment of $\mathbf{a}$ is $a(k-1)$. Then we have $a(k-1) < b(j-1)$ and $b(j) > a(k-2)$, a violation of condition (iii).

We have thus scheduled the expanding phase of $\mathbf{b}$, up to but not including the first global minimum of $\mathbf{b}$, against a section of $\mathbf{a}$ not going beyond the first global maximum of $\mathbf{a}$. We can schedule the contracting phase symmetrically. If there is a global maximum of $\mathbf{b}$ between two global minima, then we may schedule this against any global maximum of $\mathbf{a}$. It remains to schedule the global minimum or minima of $\mathbf{b}$ against the missing section(s) of $\mathbf{a}$: condition (ii) assures us that these global minima are below all entries of $\mathbf{a}$, so we can complete the schedule φ. □

The proof of this theorem also gives us a fast algorithm to determine whether two worms are comparable.

In fact, the partial order relation $\preceq$ on the set $\mathbb{W}$ of worms has the structure of a lattice. This result is due to Moseman, and a proof can be found in her doctoral thesis [24].

For information, we give here the construction of the join of two worms; the construction of the meet is similar.

The join $\mathbf{a} \vee \mathbf{b}$ is the worm equivalent to the word $\mathbf{c}$, whose i-entry $c(i)$ is given by $\max\big(a(\varphi_x(i)), b(\varphi_y(i))\big)$ for a particular schedule φ defined informally as follows.

The schedule φ starts by taking any initial descents in $\mathbf{a}$ and/or in $\mathbf{b}$, until we have reached a valley in both worms. Then the schedule proceeds by advancing twice successively in one of the words, from valley to peak to valley—we say that the schedule *crosses a peak* in $\mathbf{a}$ or $\mathbf{b}$. The peaks in the expanding phases of both worms are crossed first, in decreasing order of the valley to the left of the peak. Then the peaks in the contracting phase are crossed, in increasing order of the valley to the right of the peak.

For example, consider the worms $\mathbf{a} = \langle 1, 3, -2, 4, -3, 5\rangle$ and $\mathbf{b} = \langle 0, 1, -1, 6, -2\rangle$, both of which are in expanding phase throughout. The table below illustrates how to form the join of these two worms.

$\mathbf{a}$	1	3	-2	-2	-2	-2	-2	4	-3	5
$\mathbf{b}$	0	0	0	1	-1	6	-2	-2	-2	-2
$\mathbf{a} \vee \mathbf{b}$	1	3	0	1	-1	6	-2	4	-2	5

The final line of the table is a word equivalent to the worm $\langle 1, 3, -1, 6, -2, 5\rangle$, which is thus the join of $\mathbf{a}$ and $\mathbf{b}$.

Moseman also shows that the number of worms with range contained in $\{1, \ldots, n\}$ is exactly set as $(4^{n+1} - 3n - 4)/9$.

4. Adding worms. We now wish to argue that there is an optimal way to add two words $\mathbf{a}$ and $\mathbf{b}$. For general words, we cannot expect that this "best schedule," or even any schedule that minimizes $\max \mathbf{a} +^{\varphi} \mathbf{b}$, can be obtained without seeing all of both words: for example, suppose $\mathbf{a} = \langle 1, 2, 3, 4, 0\rangle$ and $\mathbf{b} = \langle 1, 2, 3, x\rangle$. Then if $x = 2$, we should first run through $\mathbf{a}$, but if $x = 0$, we should run through $\mathbf{b}$ first.

However, if $\mathbf{a}$ and $\mathbf{b}$ are worms, a 2-stage look-ahead suffices to design a schedule which produces a sum which not only minimizes the max, but lies below or equal to the sum obtained from any other schedule in the worm order.

In fact, it is just as easy to describe how to add several worms simultaneously; this will enable us to deduce Theorem 2.1 as well.

To this end, we define a *weight* $\eta_i(\mathbf{a})$ which codes the desirability of making a move in worm $\mathbf{a}$ when currently sitting in stage i. To wit:

1. If $a(i{+}1) < a(i)$, $\eta_i(\mathbf{a}) = \infty$.
2. If $a(i{+}2) < a(i) < a(i{+}1)$, $\eta_i(\mathbf{a}) = \frac{1}{a(i+1)-a(i)}$.
3. If $a(i{+}2) = a(i) < a(i{+}1)$, $\eta_i(\mathbf{a}) = 0$.
4. If $a(i) < a(i{+}2) < a(i{+}1)$, $\eta_i(\mathbf{a}) = \frac{-1}{a(i+1)-a(i+2)}$.
5. If $a(i) < a(i{+}1)$ and $a(i{+}1)$ is the final entry in the worm, then $\eta_i(\mathbf{a}) = -\infty$.

We now define, for any finite list $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_q$ of worms, a particular schedule χ recursively as follows: If $\chi(k) = (i_1, \ldots, i_q)$ and ℓ is the least index for which $\eta_{i_\ell}(\mathbf{a}_\ell) \geq \eta_{i_j}(\mathbf{a}_j)$ for all $j > \ell$, then $\chi(k{+}1) = (i_1, \ldots, i_{\ell-1}, i_\ell{+}1, i_{\ell+1}, \ldots, i_q)$. In other words, we advance the worm currently of highest weight, breaking ties (this will not matter) in favor of lower indexed worms. We then define the sum $\sum_{\ell=1}^{q} \mathbf{a}_\ell$ as $\sum_{\ell \in \{1,\ldots,q\}}^{\chi} \mathbf{a}_\ell$; in the case $q = 2$, we use the notation $\mathbf{a}_1 + \mathbf{a}_2 = \mathbf{a}_1 +^{\chi} \mathbf{a}_2$.

Before moving on to prove that χ is the promised optimal schedule, we will give a more intuitive description of this procedure. As before, if the schedule has reached a valley in each of the words, then it is said to *cross a peak* in one of the worms if it advances twice successively, from valley to peak to valley, in the same worm. The optimal schedule for summing worms again proceeds by crossing peaks one at a time, but the rule for which peak to cross next is quite different from that required to form the join of two worms.

In the schedule χ, we start by taking any initial descents in arbitrary order. Then we cross the peaks in the various worms in a specific sequence. The expanding phases of all worms are completed before any of the contracting phases are begun. In the expanding phase, we cross the peaks in increasing order of length of the "uphill" section; in the contracting phase, we cross the peaks in decreasing order of length of the "downhill" section. We finish with any final ascents.

For example, the table below shows how the sum of the two expanding worms $\mathbf{a} = \langle 1, 3, -2, 4, -3, 5\rangle$ and $\mathbf{b} = \langle 0, 1, -1, 6, -2\rangle$ is formed.

$\mathbf{a}$	1	1	1	3	-2	4	-3	-3	-3	5
$\mathbf{b}$	0	1	-1	-1	-1	-1	-1	6	-2	-2
$\mathbf{a}+\mathbf{b}$	1	2	0	2	-3	3	-4	3	-5	3

The final line of the table is a word equivalent to the worm $\langle 1, 2, -3, 3, -5, 3\rangle$, which is thus the sum of $\mathbf{a}$ and $\mathbf{b}$.

If $\mathbf{a} = \langle a(0), \ldots, a(m)\rangle$ is a worm where $a(0)$ is a valley, and hence all the odd values $a(2j{+}1)$ are peaks, then $\eta_{2j}(\mathbf{a})$ is finite for each j: we think of this as the "weight of the peak $a(2j{+}1)$." In the expanding phase, we have $\eta_{2j}(\mathbf{a}) = 1/(a(2j{+}1) - a(2j)) > 0$; these are the reciprocals of the lengths of the uphill sections of the successive peaks, so this sequence of positive weights is decreasing. In the contracting phase, we have $\eta_{2j}(\mathbf{a}) = -1/(a(2j{+}1) - a(2j{+}2)) < 0$, and again these numbers—which are minus the reciprocal of the length of the descent from the peak $a(2j{+}1)$—form a decreasing sequence. Thus the sequence $\eta_{2j}(\mathbf{a})$ of weights of peaks is strictly decreasing in j over the length of the worm. If $a(0)$ is a peak, then it is the sequence $\eta_{2j+1}(\mathbf{a})$ that represents the weight of the next peak $a(2j{+}2)$, and once more this sequence is strictly decreasing.

So the formal specification of our schedule χ amounts to saying that we always advance in any worm from peak to valley when we can and, when we have reached a valley in each worm, we do indeed cross the peaks in the order described above.

Our aim is to prove that, in the worm order, $\sum_\ell \mathbf{a}_\ell$ is below the sum produced by any other schedule. We start by proving this in the case where there are just two worms.

THEOREM 4.1. *For any two worms* $\mathbf{a}$ *and* $\mathbf{b}$*, and any schedule* φ*,* $\mathbf{a}+\mathbf{b} \preceq \mathbf{a}+^{\varphi}\mathbf{b}$.

Proof. Let φ be any schedule other than χ; we show that there is a modification ψ of φ which is closer to χ in a certain natural metric, and which satisfies $\mathbf{a}+^{\psi}\mathbf{b} \preceq \mathbf{a}+^{\varphi}\mathbf{b}$. Since there are only finitely many schedules, this process will terminate at χ.

Suppose φ's failure to respect weights involves overruling an infinite weight, say $\eta_i(\mathbf{a})$, by moving from $b(j)$ to $b(j{+}1)$, instead of moving from $a(i)$ to the lower value $a(i{+}1)$ at time $k = i{+}j{+}1$. We take ψ to choose to move in $\mathbf{a}$ at that point, and otherwise behave like φ until the next time k' when φ advances in $\mathbf{a}$, after which the two schedules will rejoin. But then the ℓ-entry of the word $\mathbf{a} +^{\psi} \mathbf{b}$ is less than the $(\ell{-}1)$-entry of $\mathbf{a} +^{\varphi} \mathbf{b}$ for all ℓ with $k \le \ell < k'$, and otherwise the two schedules give the same values, so $\mathbf{a} +^{\psi} \mathbf{b} \preceq \mathbf{a} +^{\varphi} \mathbf{b}$.

A similar argument shows that we may assume that any final ascents in either of the worms may be assumed to be scheduled at the very end of the sequence.

Thus φ can be assumed to cross one peak at a time, apart from initial or final moves in a worm that starts with a descent or finishes with an ascent. We measure the distance from φ to χ by the number of pairs of peaks, one from $\mathbf{a}$ and one from $\mathbf{b}$, which are scheduled out of weight order by φ. As long as $\varphi \neq \chi$ there must be a *consecutive* pair of peaks, in different worms, which are crossed out of weight order in $\mathbf{a} +^{\varphi} \mathbf{b}$; we claim that, in such a case, switching the order in which the two peaks are crossed produces a lower or equivalent subworm. Such a switch will reduce the distance from φ to χ by 1, thus proving the theorem.

To prove the claim, we may add or subtract constants so that the entries of $\mathbf{a}$ and $\mathbf{b}$ at the valleys before the pair of peaks are both 0; suppose the value in $\mathbf{a} +^{\varphi} \mathbf{b}$ then proceeds from 0 up to w, down to x, up to y and down to z, as the two peaks are crossed out of weight order. There are three cases:

1. The first of the two peaks crossed in φ has a negative weight and the second a positive weight.
   This means that $x > 0$ (the first peak is in contracting phase), but $z < x$ (the second peak is in expanding phase). Then x is greater than both 0 and z, so the portion $\langle 0, w, x, y, z\rangle$ of the word $\mathbf{a} +^{\varphi} \mathbf{b}$ contains a spanner, and collapses to $\langle 0, \max(w, y), z\rangle$. Switching the order in which the two peaks are crossed (i.e., scheduling them in weight order) results in

   $$\langle 0, y-x, z-x, w+z-x, z\rangle \preceq \langle 0, y-x, 0, w+z-x, z\rangle \equiv \langle 0, \max(y-x, w+z-x), z\rangle,$$

   which is lower since $y-x < y$ and $w+z-x < w$. This argument also covers the case where one of the weights is zero.
2. Both weights are positive, but that of the first peak crossed is lower.
   This means that both peaks are in expanding phase, i.e., $0 > x > z$, but $y-x < w$. Then $y < w+x < w$ so the section of the sum collapses to $\langle 0, w, z\rangle$. Switching the order in which the peaks are crossed again yields $\langle 0, y-x, z-x, w+z-x, z\rangle \preceq \langle 0, \max(y-x, w+z-x), z\rangle$ and both $y-x$ and $w+z-x$ are less than w.
3. Both weights are negative, but that of the first peak crossed is lower.
   This means that both peaks are in contracting phase, i.e., $0 < x < z$ and $z < x$, but $y-z > w-x$. This is merely a right-to-left reversal of the previous case, shifted up by z.

This concludes the proof of Theorem 4.1. □

With the help of the last statement of Lemma 3.1, Theorem 4.1 tells us that addition of worms is associative—in fact, the set of worms forms a lattice-ordered semigroup, with $\mathbf{0}$ as an identity; see [24]. We now have the following result.

THEOREM 4.2. *For any list* $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_q$ *of worms, and any schedule* φ*,*

$$\sum_{\ell=1}^{q} \mathbf{a}_\ell \preceq \sum_{\ell\in\{1,\ldots,q\}}^{\varphi} \mathbf{a}_\ell.$$

From this, Theorem 2.1 follows. Indeed, notice that the order in which we cross the peaks in any two worms does not depend on what other worms are involved in the sum.

Let us be slightly more precise about the time-complexity of scheduling q words of total length n. For convenience, we measure length simply according to the number

of entries, and assume that adding or comparing entries can be carried out in unit time.

Given a single word of length n_ℓ, we can calculate its worm by reading from the front, noting any "records" of highest or lowest values, and then doing the same while reading the worm backwards. This can be done in time $O(n_\ell)$, so all worms can be found in time $O(n)$.

Given q worms of total length at most n, we calculate the schedule optimizing their sum as follows. First, in linear time, we find all the weights $\eta_i(\mathbf{a}_\ell)$, and write these as q lists of weights. Throughout the process, we maintain a sorted list L of the weights at the current point of each of the q lists. A step consists of taking the maximum entry, say $a_\ell(i)$, in L, and then inserting the next weight $\eta_{i+1}(\mathbf{a}_\ell)$ into L using binary insertion sort. Each step thus takes time $O(\log q)$, and the total length of time required to construct the schedule is $O(n \log q)$. (Of course, we really only have to consider one weight for each peak, and this part of the problem is then exactly equivalent to that of merging q sorted lists.)

Once we have the schedule for the worms, we can calculate the worm-sum, and/or schedule the original words, in linear time.

Thus, overall, the problem of optimally scheduling the words can be solved in time $O(n \log q)$. At least in our model of computation, this is best possible, since the problem includes that of summing $q = n/2$ worms, each consisting of a single valley-peak-valley sequence, and this is equivalent to sorting the $n/2$ weights.

5. Chains in a submodular system. In this section we will generalize the lattice paths to maximal chains in a modular lattice, and the sums of real words to values of a submodular function.

A lattice $\mathcal{D}$ is *distributive* if for every A, B, and $C \in \mathcal{D}$, $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$. Equivalently, $\mathcal{D}$ is distributive if it contains neither of the two lattices $\mathcal{C}_5$ or $\mathcal{M}_3$ pictured below in Figure 3 as a sublattice.

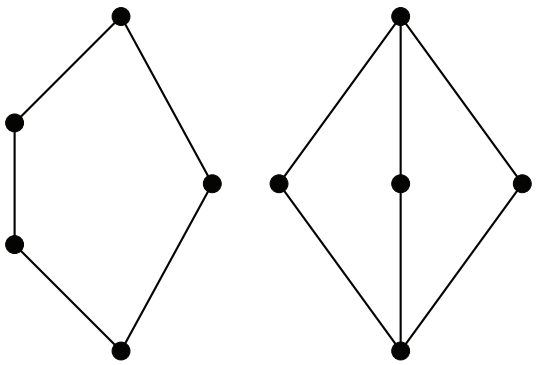

FIG. 3. *The lattices $\mathcal{C}_5$ (left) and $\mathcal{M}_3$ (right).*

The lattices we consider will be finite, hence endowed with a minimum element which we call 0, and a maximum element 1. Every distributive lattice $\mathcal{D}$ is the lattice of ideals (downward-closed subsets) of a poset (partially ordered set), and conversely. We can always take the poset to be the set of join-irreducibles of $\mathcal{D}$, that is, the set of elements other than 0 which cannot be obtained as the join of two other elements.

A real-valued function f on a distributive lattice $\mathcal{D}$ is said to be *modular* if

$$f(A \wedge B) + f(A \vee B) = f(A) + f(B)$$

for every A, $B \in \mathcal{D}$, and *submodular* if

$$f(A \wedge B) + f(A \vee B) \leq f(A) + f(B).$$

A distributive lattice $\mathcal{D}$ together with a submodular function on $\mathcal{D}$ is called a *submodular system* (cf. [12]).

Submodular functions on distributive lattices are important in optimization (see, e.g., [8, 12, 29]); indeed for $\mathcal{D} = [m] \times [n]$ they are often called Monge arrays and were studied as far back as 1781 [23].

In any lattice, a *maximal chain*, or *maxchain* for short, is a sequence $\mathcal{C} = \langle C_0, C_1, \ldots, C_n \rangle$ with $0 = C_0 < C_1 < \cdots < C_n = 1$, such that each (C_i, C_{i+1}) is a *covering pair*, i.e., there is no B such that $C_i < B < C_{i+1}$. A *path* from 0 to 1 in a lattice is a sequence $\mathcal{P} = \langle P_0, P_1, \ldots, P_m \rangle$, with $P_0 = 0$ and $P_m = 1$, in which each pair (P_i, P_{i+1}) is a covering pair in one direction or the other. This coincides with the usual notion of a path in the graph that is the (undirected) Hasse diagram of the lattice. Sometimes we allow repeats on our paths and maxchains; i.e., we allow $P_i = P_{i-1}$ or $C_i = C_{i-1}$ for some values of i. This makes no real difference as we can always consider the path/chain derived by suppressing the repeats.

Given a path (or chain) $\mathcal{P} = \langle P_0, P_1, \ldots, P_m \rangle$ in $\mathcal{L}$, we set $\mathcal{P}[P_i, P_j]$ to be the subsequence $\langle P_i, \ldots, P_j \rangle$, for $i \leq j$.

In fact our results apply to a more general class than the distributive lattices, namely the class of *modular* lattices. A (finite) lattice $\mathcal{L}$ is modular if every maxchain is the same length, and the rank function is modular. Equivalently, $\mathcal{L}$ is modular if it does not contain the lattice $\mathcal{C}_5$ as a sublattice.

Given a path $\mathcal{P} = \langle P_0, P_1, \ldots, P_m \rangle$ from P_0 to P_m in a modular lattice $\mathcal{L}$, and an element $B \in \mathcal{L}$, we define $\mathcal{P} \wedge B$ to be the sequence obtained from $\langle P_0 \wedge B, \ldots, P_m \wedge B \rangle$ by collapsing any consecutive repeated elements. The sequence $\mathcal{P} \vee B$ is defined analogously. The key fact we need about modular lattices is that $\mathcal{P} \wedge B$ is a path from $P_0 \wedge B$ to $P_m \wedge B$, and $\mathcal{P} \vee B$ is a path from $P_0 \vee B$ to $P_m \vee B$.

The lattice of ideals of the poset consisting of the union of disjoint linear orders of length $m_1, \ldots, m_q$ is the distributive lattice $[m_1] \times \cdots \times [m_q]$. If $\mathbf{a}_1, \ldots, \mathbf{a}_q$ are words of length $m_1, \ldots, m_q$, then the function f given by $f(x_1, \ldots, x_q) = \sum_{j=1}^{q} a_j(x_j)$ is modular. Conversely, given any modular function f on $\prod_{j=1}^{q} [m_j]$, there are words which generate it as above, and they are unique modulo shifts of the words by any sequence of constants $c_1, \ldots, c_q$ summing to 0.

For a function f on a lattice $\mathcal{L}$, and any sequence $\mathcal{P} = \langle P_0, P_1, \ldots, P_n \rangle$ of elements of $\mathcal{L}$, let $f(\mathcal{P})$ denote the word $\langle f(P_0), f(P_1), \ldots, f(P_n) \rangle$. The worm equivalent to $f(\mathcal{P})$ will be called the *worm of* $\mathcal{P}$, where the function f is understood to be fixed.

Theorem 4.2 says in effect that, for any modular function f on $\prod_{j=1}^{q} [m_j]$, there is a maxchain $\mathcal{C}$ such that, for any other maxchain $\mathcal{D}$, we have $f(\mathcal{C}) \preceq f(\mathcal{D})$. The following is thus a generalization of Theorem 4.2.

THEOREM 5.1. *For any submodular function f on any modular lattice $\mathcal{L}$, there is a maxchain $\mathcal{C}$ such that, for any other maxchain $\mathcal{D}$, $f(\mathcal{C}) \preceq f(\mathcal{D})$.*

We need some notation, and one lemma, in order to prove this result.

We take an arbitrary modular lattice $\mathcal{L}$, submodular function f, and maxchain $\mathcal{C}$ from 0 to 1. For $M \in \mathcal{L}$, let $M^{\mathcal{C}} = \min\{C_i \in \mathcal{C} : \ C_i \geq M\}$ and $M_{\mathcal{C}} = \max\{C_i \in \mathcal{C} : \ C_i \leq M\}$.

An element $M \in \mathcal{L} \setminus \mathcal{C}$ will be called a *lure* (for $\mathcal{C}$) if $f(M) \leq f(A)$ for all A with $M_{\mathcal{C}} \leq A < M$ or $M < A \leq M^{\mathcal{C}}$.

LEMMA 5.2. *If M is a lure for a maxchain $\mathcal{C}$, then there is a maxchain $\mathcal{C}'$ with $f(\mathcal{C}') \preceq f(\mathcal{C})$ that agrees with $\mathcal{C}$ below $M_{\mathcal{C}}$ and above $M^{\mathcal{C}}$, and passes through M.*

Proof. Given a maxchain $\mathcal{C}$, and a lure M for $\mathcal{C}$, we set $\mathcal{D}$ to be the section $\mathcal{C}[M_{\mathcal{C}}, M^{\mathcal{C}}]$ of $\mathcal{C}$, and define $\mathcal{D}' = (\mathcal{D} \wedge M) \mid (\mathcal{D} \vee M)$, the concatenation of the chain $\mathcal{D} \wedge M$ from $M_{\mathcal{C}}$ to M, and the chain $\mathcal{D} \vee M$ from M to $M^{\mathcal{C}}$. Then we define the maxchain

$$\mathcal{C}' := \mathcal{C}[0, M_{\mathcal{C}}] \mid \mathcal{D}' \mid \mathcal{C}[M^{\mathcal{C}}, 1].$$

To complete the proof, it suffices to show that $f(\mathcal{D}') \preceq f(\mathcal{D})$, as this implies that $f(\mathcal{C}') \preceq f(\mathcal{C})$.

For each D_i in $\mathcal{D}$, observe that $f(D_i \vee M) \geq f(M)$ as M is a lure, and that $f(D_i \wedge M) + f(D_i \vee M) \leq f(D_i) + f(M)$ by submodularity, so $f(D_i \wedge M) \leq f(D_i)$. Similarly $f(D_i \vee M) \leq f(D_i)$, since $f(D_i \wedge M) \geq f(M)$.

Now let D_j maximize $f(D_j)$ over $\mathcal{D}$. We schedule $D_i \wedge M$ opposite D_i until $i = j$, then continue in $\mathcal{D}'$ as far as $D_j \vee M$, and then schedule $D_i \vee M$ opposite D_i thereafter. This schedule demonstrates that $f(\mathcal{D}') \preceq f(\mathcal{D})$, as claimed. ∎

We now turn to the proof of Theorem 5.1.

Proof. It turns out to be convenient to be able to assume that $f(A) \neq f(B)$ whenever A and B are distinct elements of $\mathcal{L}$. Indeed, suppose we can prove the result under this assumption. Given any submodular function f on $\mathcal{L}$, we take another submodular function g on $\mathcal{L}$ such that $g(A) \neq g(B)$ for distinct A and B—such functions are easy to construct—and consider $f + \varepsilon g$, where $\varepsilon > 0$ is chosen small enough that, whenever $f(A) > f(B)$, then also $f(A) + \varepsilon g(A) > f(B) + \varepsilon g(B)$. Applying the theorem to $f + \varepsilon g$, which does take distinct values, we obtain a maxchain $\mathcal{C}$ such that $(f + \varepsilon g)(\mathcal{C}) \preceq (f + \varepsilon g)(\mathcal{D})$ for all maxchains $\mathcal{D}$. Then also $f(\mathcal{C}) \preceq f(\mathcal{D})$ for all $\mathcal{D}$.

From now on, we do indeed assume that f takes distinct values on elements of $\mathcal{L}$.

Let A be the location of the global minimum of f over $\mathcal{L}$. For any maxchain $\mathcal{D}$, either $\mathcal{D}$ passes through A, or A is a lure for $\mathcal{D}$, in which case, by Lemma 5.2, there is a maxchain through A that is worm-below, or worm-equivalent, to $\mathcal{D}$ (in fact, the former). Thus, in proving the theorem, we may restrict attention to maxchains passing through A.

We shall describe a specific maxchain in the sublattice below A and show that any maxchain in this sublattice is worm-below or worm-equivalent to this chain. If we can do this, then we can clearly do the same thing above A, and this will complete our proof. Thus, for convenience, we may as well assume that $A = 1$, the top point of the lattice $\mathcal{L}$.

All worms of maxchains $\mathcal{C}$ in $\mathcal{L}$ are entirely in expanding phase, and the maximum value of such a worm is its penultimate entry. Among all maxchains in $\mathcal{L}$, we choose $\mathcal{C}$ to be one with the lexicographically least worm, reading backwards from $f(1)$ to $f(0)$. In particular, this choice guarantees that the maximum value of $f(\mathcal{C})$ is the minimum among all maxchains in $\mathcal{L}$.

We claim that every maxchain $\mathcal{D}$ in $\mathcal{L}$ has $f(\mathcal{C}) \preceq f(\mathcal{D})$. Suppose not, and let $\mathcal{D}$ be a maxchain such that $f(\mathcal{C}) \not\preceq f(\mathcal{D})$.

In what follows, if $f(B_i)$ is a peak/valley in the worm of a maxchain $\mathcal{B}$, we let B_i^+ be the location of the next valley/peak above B_i on the worm, and B_i^- the location of the next valley/peak below B_i.

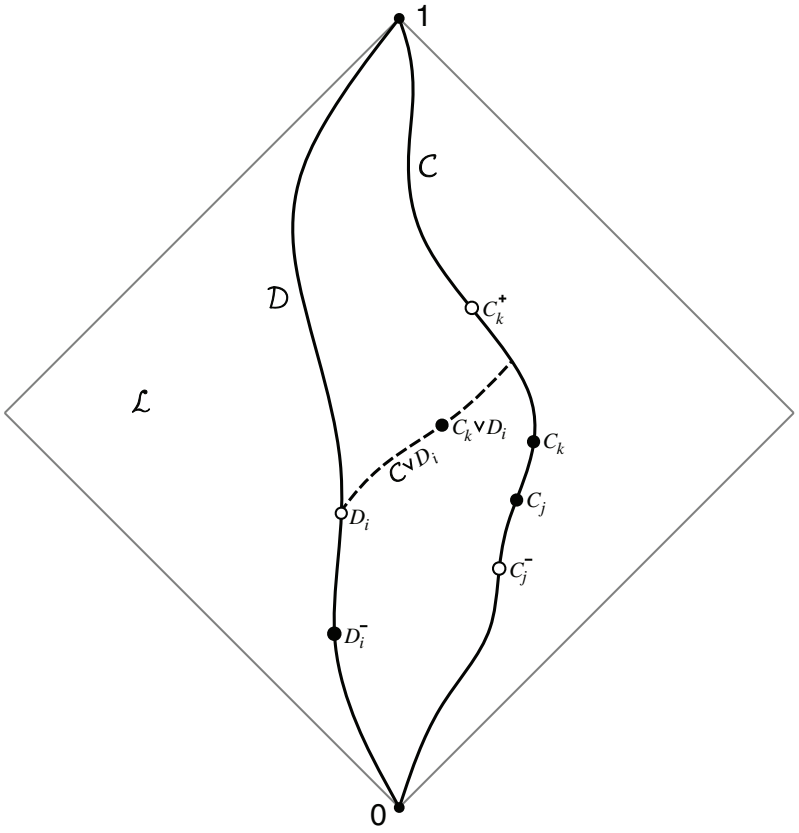

FIG. 4. *Points in $\mathcal{L}$ from the proof of Theorem* 5.1.

Using Theorem 3.3, we see that there is a peak C_j of the worm of $\mathcal{C}$ and a valley D_i of the worm of $\mathcal{D}$ such that (a) $f(C_j) > f(D_i^-)$, and (b) $f(D_i) < f(C_j^-)$. See Figure 4 for a rough sketch of the situation described here and below.

Now consider the minimum of f on the sublattice below D_i. If this is attained at some point $D < D_i$, then D is a lure for the section $\mathcal{B}$ of the chain $\mathcal{D}$ below D_i, so by Lemma 5.2 there is a maxchain $\mathcal{B}'$ from 0 to D_i with $f(\mathcal{B}') \preceq f(\mathcal{B})$. Thus the highest peak on $f(\mathcal{B}')$ has height at most $f(D_i^-) < f(C_j)$, and the valley above this in $f(\mathcal{B}')$ is D, with $f(D) < f(D_i) < f(C_j^-)$. What this means is that, replacing $\mathcal{B}$ by $\mathcal{B}'$ if necessary, we may assume that $f(D) > f(D_i)$ whenever $D < D_i$.

Consider now the maxchain $\mathcal{C}' = \mathcal{D}[0, D_i] \mid \mathcal{C} \vee D_i$: we claim that $f(\mathcal{C}')$ is lexicographically lower than $f(\mathcal{C})$, which will be a contradiction.

If $D_i < C_j$, then $\mathcal{C}'$ agrees with $\mathcal{C}$ at and above C_j, but goes through D_i: as $f(D_i) < f(C_j^-)$, which is the lowest value taken by f on the portion of $\mathcal{C}$ below C_j, this maxchain is indeed lexicographically strictly below $\mathcal{C}$. Thus $D_i \not< C_j$.

For $C_k \in \mathcal{C}$, we have $f(C_k \wedge D_i) \geq f(D_i)$, and therefore $f(C_k \vee D_i) \leq f(C_k)$, with strict inequality unless $C_k > D_i$.

Let ℓ be the largest value such that $f(C_\ell)$ is a peak or valley of the worm of $\mathcal{C}$, and $C_\ell \not> D_i$; observe that $\ell \geq j$. If $f(C_\ell)$ is a valley, we see immediately that $f(\mathcal{C}')$ is

lexicographically below $f(\mathcal{C})$, as it takes in the lower value $f(C_\ell \vee D_i)$. Thus $f(C_\ell)$ is a peak. In this case, we again have that $f(C_\ell \vee D_i) < f(C_\ell)$, but for our contradiction we also need to prove that f does not assume a value as high as $f(C_\ell)$ anywhere else on the section of $\mathcal{C}'$ below C_ℓ^+.

We have $f(C_k \vee D_i) \leq f(C_k) < f(C_\ell)$ for all $C_k \neq C_\ell$ below C_ℓ^+ on $\mathcal{C}$, and additionally we have $f(D_k) \leq f(D_i^-) < f(C_j) \leq f(C_\ell)$ for all D_k on the portion of $\mathcal{C}'$ below D_i. Therefore $f(\mathcal{C}')$ does indeed stay under $f(C_\ell)$, and we have our contradiction. □

Remark. Note that Theorem 5.1 fails for *any* nonmodular lattice, as can be seen by assigning $f(0) = f(1) = 0$, $f(A) = 1$, $f(B) = -1$, and $f(C) = 2$ in the lattice $\mathcal{C}_5$, with $0 < A < 1$ and $0 < B < C < 1$, and observing that $\langle 0, 1, 0\rangle$ and $\langle 0, -1, 2, 0\rangle$ are incomparable worms.

6. Maxchains versus paths. Now that we know there is a worm-minimum maxchain in any submodular system, we can ask whether this chain in fact beats any *path* from 0 to 1 in our modular lattice.

We shall prove that, for any submodular function f on a modular lattice $\mathcal{L}$, and any path $\mathcal{P} = \langle P_0, P_1, \ldots, P_m\rangle$ from 0 to 1 in $\mathcal{L}$, there is a maxchain $\mathcal{C} = \langle C_0, C_1, \ldots, C_m\rangle$, with repeats allowed, such that $f(C_i) \leq f(P_i)$ for $i = 0, \ldots, m$. In other words, the maxchain $\mathcal{C}$ "beats" the path $\mathcal{P}$ pointwise, so certainly $f(\mathcal{C}) \preceq f(\mathcal{P})$.

In fact, we shall prove something slightly stronger, as we now describe.

A *slipchain* in a modular lattice $\mathcal{L}$ is a sequence $\mathcal{P} = \langle P_0, P_1, \ldots, P_m\rangle$ of elements of $\mathcal{L}$, with $P_0 = 0$ and $P_m = 1$, such that, for each $i = 1, \ldots, m$, $P_i \leq U_i$ for some upper cover U_i of P_{i-1}. Clearly every path from 0 to 1 in a lattice is a slipchain.

THEOREM 6.1. *Let f be a submodular function on a modular lattice $\mathcal{L}$. Let $\mathcal{P} = \langle P_0, P_1, \ldots, P_m\rangle$ be any slipchain in $\mathcal{L}$. Then there is a maxchain $\mathcal{C} = \langle C_0, C_1, \ldots, C_m\rangle$, with repeats allowed, such that $f(C_i) \leq f(P_i)$ for $i = 0, \ldots, m$.*

Bienstock and Seymour [3] prove a special case of this result (their (2.2)) for the case where $\mathcal{L}$ is a Boolean algebra (their result is stated only for one specific class of submodular functions—see subsection 8.4—but submodularity is all they use). In fact, they effectively proved Corollary 6.3 for the power-set lattice. Bienstock and Seymour also note that their result can be seen as a special case of Theorem 3.2 of Robertson and Seymour [28].

The Bienstock–Seymour proof generalizes without too much difficulty to the setting of a general modular lattice; the proof we give has a flavor similar to theirs, but with some differences to accommodate the more general setting.

Proof. For $j = 1, \ldots, m$, set $Q_j = \bigvee_{i=0}^{j} P_i$. If $f(Q_j) \leq f(P_j)$ for all j, then we claim that setting $C_j = Q_j$ suffices. To see this, observe that, for each j, either $P_j \leq Q_{j-1}$, in which case $Q_j = Q_{j-1}$, or there is some upper cover U_j of P_{j-1} such that $P_j \leq U_j$, and U_j is not below Q_{j-1}. In the latter case, $U_j \wedge Q_{j-1} = P_{j-1}$, and so U_j covers $U_j \wedge Q_{j-1}$; the modularity property now implies that $U_j \vee Q_{j-1}$ covers Q_{j-1}. As $Q_j = P_j \vee Q_{j-1} \leq U_j \vee Q_{j-1}$ and Q_j is above Q_{j-1}, we must have $Q_j = U_j \vee Q_{j-1}$, a cover of Q_{j-1} as desired.

Therefore we may assume that there is some j such that $f(Q_j) > f(P_j)$. Now choose D with $P_j \leq D < Q_j$ to be maximal in this interval subject to $f(D) < f(Q_j)$.

Consider the sequence $\langle P_0', \ldots, P_j'\rangle$, where $P_i' = P_i \wedge D$ for each $i = 0, \ldots, j$. This is a slipchain from 0 to $P_j \wedge D = P_j$. For $i = 0, \ldots, j$, as $D \leq P_i \vee D \leq Q_j$, we have $f(P_i \vee D) \geq f(D)$, and therefore by submodularity we have $f(P_i') = f(P_i \wedge D) \leq f(P_i)$. We cannot have $P_i = P_i \wedge D$ for all $i = 0, \ldots, j$, as then all the P_i would be below D and hence $D \geq \bigvee_{i=0}^{j} P_j = Q_j$.

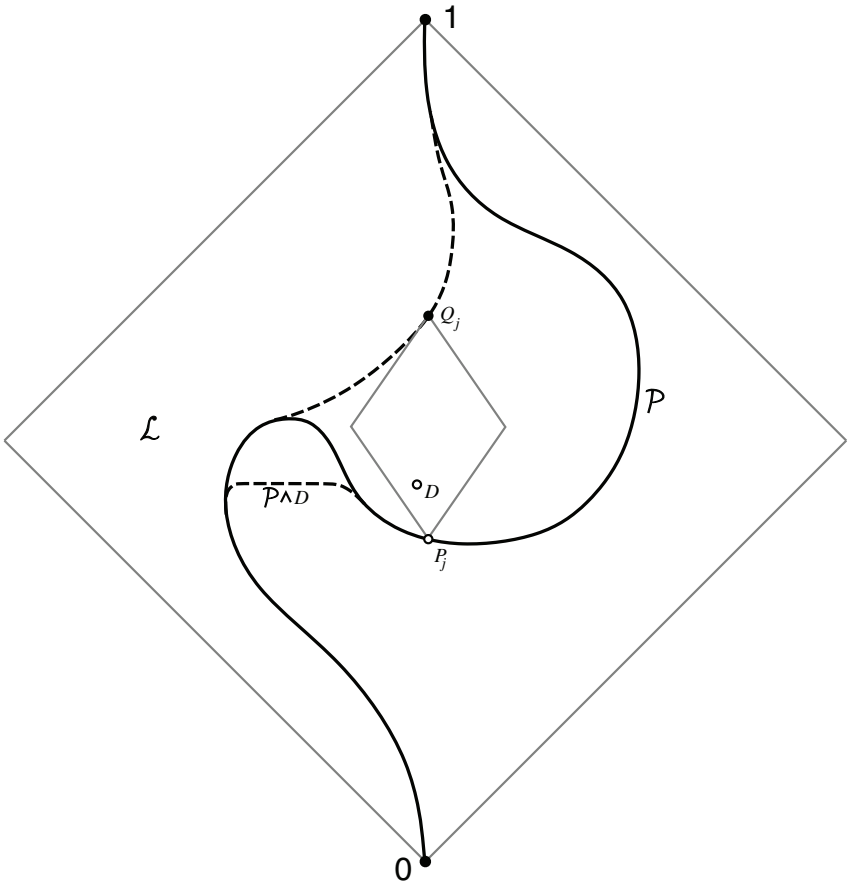

FIG. 5. *Shortening a slipchain in the proof of Theorem* 6.1.

If we further set $P'_i = P_i$ for $i \geq j$, we obtain a "strictly lower" slipchain $\mathcal{P}' = \langle P'_0, \ldots, P'_m \rangle$—meaning that $P'_i \leq P_i$ for all $i = 0, \ldots, m$, with strict inequality for at least one i—such that $f(P'_i) \leq f(P_i)$ for each $i = 0, \ldots, m$.

See Figure 5 for a schematic picture of the proof.

If $\mathcal{P}'$ is not a maxchain, we repeat the process to obtain either a maxchain or a strictly lower slipchain whose f-values are pointwise no greater than those on $\mathcal{P}'$. We cannot continue indefinitely in this way, and so we must eventually finish with a slipchain that is a maxchain. □

THEOREM 6.2. *Let f be a submodular function on a modular lattice $\mathcal{L}$. Then there is a maxchain $\mathcal{C} = \langle C_0, C_1, \ldots, C_n \rangle$ in $\mathcal{L}$ such that for any slipchain $\mathcal{P} = \langle P_0, P_1, \ldots, P_m \rangle$ from* 0 *to* 1 *in $\mathcal{L}$, $f(\mathcal{C}) \preceq f(\mathcal{P})$.*

Proof. The statement follows from Theorems 5.1 and 6.1, together with the fact that eliminating repeated values from a maxchain with repetitions yields worm-equivalent words. □

COROLLARY 6.3. *If there is a slipchain $\mathcal{P}$ from* 0 *to* 1 *in a submodular system in which every point P on $\mathcal{P}$ has $f(P) \leq b$, then there is a maxchain with the same*

property.

Using a variation of the proof of Theorem 6.1, we can prove a slightly stronger form of this useful corollary. Let $\mathcal{S}$ be a subset of a modular lattice $\mathcal{L}$; a *bone* in $\mathcal{S}$ is a pair A, B of elements of $\mathcal{S}$ such that neither $A \vee B$ nor $A \wedge B$ is in $\mathcal{S}$. If there is no such pair, we say $\mathcal{S}$ is "boneless"; the set $\mathcal{S} = \{A \in \mathcal{L} : \ f(A) \leq b\}$, for any submodular function f on $\mathcal{L}$, is an example of a boneless set. We have the following result.

PROPOSITION 6.4. *Let $\mathcal{S}$ be a boneless set in a modular lattice $\mathcal{L}$. If $\mathcal{S}$ contains a slipchain from 0 to 1, then it contains a maxchain.*

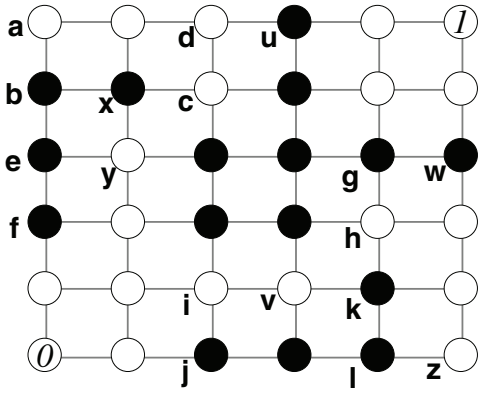

FIG. 6. *A "boneless" set not obtainable from a submodular function.*

Proof. We follow the proof of Theorem 6.1 closely. (The proof of Bienstock and Seymour [3] can also be adapted to prove this result.)

Let $\mathcal{P} = \langle P_0, P_1, \ldots, P_m \rangle$ be a slipchain in $\mathcal{L}$ with all the P_i lying in $\mathcal{S}$. As before, let $Q_j = \bigvee_{i=0}^{j} P_i$ for $j = 1, \ldots, m$. If all the Q_j lie in S, then $\mathcal{Q} = \langle Q_0, Q_1, \ldots, Q_m \rangle$ is a maxchain in $\mathcal{S}$, as required.

If some Q_j does not lie in $\mathcal{S}$, then we choose D with $P_j \leq D \leq Q_j$ to be a maximal element of $\mathcal{S}$ in this interval. For each $i = 0, \ldots, j$, we see that both P_i and D are in $\mathcal{S}$, but $P_i \vee D$ is not. Therefore, since $\mathcal{S}$ is boneless, $P_i \wedge D \in \mathcal{S}$ for each $i = 0, \ldots, j$.

As before, the sequence $\mathcal{P}' = \langle P_0 \wedge D, \ldots, P_j \wedge D = P_j, \ldots, P_m \rangle$ is a slipchain lying inside $\mathcal{S}$, and $\mathcal{P}'$ is strictly lower than $\mathcal{P}$. □

We remind the reader that all paths are slipchains, so all the results of this section are valid when $\mathcal{P}$ is a path.

Proposition 6.4 really is stronger than Corollary 6.3 in the sense that there are distributive lattices $\mathcal{D}$ with boneless subsets $\mathcal{S}$ which are not equal to $\{A \in \mathcal{D} : \ f(A) \leq 0\}$ for any submodular function f. In Figure 6, the white discs represent "open" points, that is, points in $\mathcal{S}$, and the black discs closed points, in the lattice $[5] \times [5]$. The boldface letters represent the values of some presumed submodular function at the indicated points. By applying submodularity to various rectangles we see that $b+d \leq a+c$, $e+x \leq b+y$, $f+g \leq e+h$, $i+u \leq d+v$, $j+k \leq i+\ell$, and $\ell+w \leq g+z$. Summing and cancelling gives

$$f + x + u + k + w + j \leq a + c + y + h + v + z,$$

but the six left-hand values are all for closed points and the six right-hand values for open points, so it cannot be that the open points are exactly those for which the submodular function lies below some bound.

It is less surprising (but equally true) that there is a distributive lattice $\mathcal{D}$ with a subset $\mathcal{S}$ where $\mathcal{S}$ and $\mathcal{D} \setminus S$ are *both* boneless, yet there is no *modular* function f on $\mathcal{D}$ for which $\mathcal{S} = \{A \in \mathcal{D} : \ f(A) \leq 0\}$. For example, we can take $\mathcal{D}$ to be the lattice of subsets of $\{1, 2, 3, 4\}$, and $\mathcal{S} = \{\varnothing, \{1\}, \{2\}, \{3\}, \{4\}, \{1, 2\}, \{3, 4\}\}$.

7. Complexity issues for percolation. The proof of Theorem 5.1 might appear at first glance to give an algorithm for finding a maxchain $\mathcal{C}$ such that $f(\mathcal{C})$ is minimal in the worm order. However, there are two tasks we have to perform in order to construct the "greedy" chain $\mathcal{C}$ as in the proof of that theorem. One is to minimize the submodular function f over the lattice $\mathcal{L}$, and the second is to find a chain $\mathcal{C}$ such that $\max f(\mathcal{C})$ is minimized. (In most applications, the second problem is in any case the main concern.)

When the lattice $\mathcal{D}$ is distributive, it can be efficiently presented by giving a poset X and identifying $\mathcal{D}$ with its lattice of ideals. The problem of minimizing a submodular function over a distributive lattice $\mathcal{D}$ presented in this way has been much studied—see Chapter 10 of the book of Grötschel, Lovász, and Schrijver [17] for a full account—and it is well-known that there is a polynomial time algorithm to find the minimum, when the submodular function is accessed via an oracle.

However, the second task is more problematic. What we now show is that, even in a very restrictive setting, where we are given an explicit formula for a *modular* function on a distributive lattice, the problem of deciding whether there is a chain $\mathcal{C}$ such that $f(C) \leq b$ for all $C \in \mathcal{C}$ is NP-complete. As we explain in subsection 8.4, it is also possible to deduce this from a result of Megiddo et al. [22].

When f is a modular function on a distributive lattice $\mathcal{D}$, an instance of the problem can be described economically and naturally, which thus makes it especially interesting from a complexity point of view. Specifically, we assume that we are given f and $\mathcal{D}$ in the form of a poset X, with weights $w : \ X \to \mathbb{R}$: as before, $\mathcal{D}$ is the lattice of ideals, and the modular function f is defined by $f(I) := \sum_{x \in I} w(x)$. We then wish to find a sequence $(I_0, I_1, \ldots, I_m)$ of ideals, beginning with $\emptyset$ and ending with X itself, where $|I_i \triangle I_{i-1}| = 1$ for each $i > 0$, and each $f(I_i) \leq b$ for some given bound b. If there is such a sequence, we say that X (with its weight function) *percolates*.

Corollary 6.3 tells us that in this problem, which we call PERCOLATION OF IDEALS, there is a solution which is a chain if there is any solution at all. Maximal chains of ideals correspond to linear extensions of the poset X: the ideals in a chain are the initial segments of the extension.

Formally then, the input to an instance of the problem PERCOLATION OF IDEALS is a poset X with a weight function w, and a bound b. The question is whether there is a linear extension $L = (x_1, x_2, \ldots, x_n)$ of X such that $\sum_{i=1}^{k} w(x_i) \leq b$ for each $k \leq n$.

It is clear that this problem lies in the class NP.

We next observe that PERCOLATION OF IDEALS can be reduced to what would seem at first glance to be a restrictive special case. However, we will then show that this special case is already NP-complete.

PROPOSITION 7.1. *Any instance of* PERCOLATION OF IDEALS *is equivalent to one in which the poset is graded with height* 2 *(i.e., every maximal chain has two elements), every minimal element has positive weight, and every maximal element has negative weight.*

Proof. Let X be an arbitrary finite poset with weight function w and bound b. Put $X' := \{x \in X : \ w(x) \neq 0\}$ with $x < y$ in X' when $x < y$ in X and $w(x) > 0 > w(y)$; then X' is a poset of height 2. We claim that X percolates if and only if X' does.

Suppose first that X percolates. Then there is a linear extension $(x_1, x_2, \ldots, x_n)$ of X with the property that $\sum_{i=1}^{k} w(x_i) \leq b$ for all k, $0 \leq k \leq n$. Dropping the elements of weight 0 yields a linear extension of X' with the same property.

On the other hand, suppose L is a linear extension of X' satisfying the bound on partial sums. Let $a(L)$ be the number of pairs x, y with x negative (i.e., having negative weight, or being maximal in X'), y positive (minimal), and $x < y$ in L; and let $b(L)$ be the number of "bad pairs" (u, v) with $u < v$ in L but $u > v$ in X. Choose L to maximize $a(L)$ and, given that, to minimize $b(L)$.

Now let (u, v) be a bad pair in which u and v are as close as possible in L. Either v is negative or u is positive; we suppose the former—the other case is symmetric. Consider the linear ordering L' of X' obtained by moving v to the position just before u in L. Then L' is again a linear extension of X', because if this results in moving v before a positive element y where $y < v$ in X, then u was already before y and of course $y < v < u$ in X implies $y < u$, which is not possible since L is a linear extension of X'.

It follows also that if v passed any positive y's on its way forwards, then $a(L') > a(L)$, a contradiction. But also v cannot pass any w with $v > w$ in X, for then (w, v) would have been a closer bad pair than (u, v). Since at least one bad pair has been eliminated and no new ones created, $b(L') < b(L)$ and we again have a contradiction.

We conclude that L has no bad pairs, but then we can reintroduce the 0-weight elements of X in appropriate places to get a linear extension of X satisfying the partial sum bound.

It remains only to note that if there are isolated points in X', then we can assume that L begins with the negative ones and ends with the positive ones; thus eliminating them from X' and changing the bound b appropriately results in an equivalent problem in which the poset is graded as well as satisfying the other criteria of the theorem. □

If we think of the relations of the partial order as being constraints on the order in which we can take the elements, then the proof amounts to saying that the only constraints that matter are those saying that we have to take a certain positive-weight element before a certain negative-weight element.

THEOREM 7.2. *The problem* PERCOLATION OF IDEALS *is NP-complete, even when the poset is graded with height* 2*, all minimal elements have positive weight and all maximal elements have negative weight.*

Proof. We will proceed by reducing the problem CLIQUE to this special case of PERCOLATION OF IDEALS. As CLIQUE is an NP-complete problem (see, e.g., [15]), this suffices to prove the result.

Given an instance of CLIQUE, i.e., a graph G with vertex set $[n]$ and a natural number k, we construct a partial order X with a weight function w, and an integer b, so that there is a chain of ideals all with weight at most b if and only if there is a k-clique in G.

The construction is very simple. Our partial order has n minimal elements $x_1, \ldots, x_n$, each of weight $(k - \frac{1}{2})N$, for some (even) $N \geq n^2$, and $\binom{n}{2}$ maximal elements, one above each pair of minimal elements. If ij is an edge of the graph G, then the maximal element above x_i and x_j has weight $-(N+1)$, while other maximal

elements have weight $-N$. We set

$$b = \frac{k^2+2k-1}{2}N - \binom{k}{2}.$$

Any contender for an optimal chain must proceed by taking in the minimal elements one by one; after the jth minimal element is added to the current down-set, the $j-1$ maximal elements above this new minimal and an earlier one may be included freely. The net gain in weight is $(k-\frac{1}{2})N - (j-1)N - \ell$ for some $\ell \in [0, j-1]$. This is positive for $j \leq k$, and negative for $j > k$. So the highest-weight down-set we encounter is at the point where we add the $(k+1)$st minimal, and the weight at this point is

$$(k+1)\left(k-\frac{1}{2}\right)N - \binom{k}{2}N - m = \frac{k^2+2k-1}{2}N - m,$$

where m is the number of maximal elements of weight $-(N+1)$ we have managed to include along with the first k minimals. Therefore this highest-weight down-set has weight as small as b if and only if the first k minimals correspond to a clique in G.

This completes the proof. □

8. Applications. We have seen from Theorem 6.2 that the key elements of a system to which our results can be applied are a distributive (or at least modular) lattice and a submodular function defined on it. Thus, for example, in the upgrade problem mentioned earlier, the possible states of the system are points in a product of chains (each chain representing the upgrade status of a component). This provides the distributive lattice, and it remains to insist that the performance function be submodular; this seems to be a reasonable assumption, requiring for example that no two components work only when exactly one is upgraded.

Indeed, if all we are interested in is keeping the system performing to a specified standard, we can instead apply Proposition 6.4, and obtain the following result.

PROPOSITION 8.1. *Suppose that, for all states of the system below the threshold standard, and all pairs of upgrades to disjoint sets of components, both of which put the system above the threshold standard, making both upgrades also puts the system above the threshold standard. Suppose also that there is some procedure, at each stage making either a single-step upgrade to one component, or an arbitrary downgrade, that moves from the initial state to the fully upgraded state while keeping the system above the threshold standard at all times. Then there is such a procedure that only makes upgrades.*

Indeed, the hypothesis above is exactly that the set of acceptable states of the system is boneless.

The next example, suggested to the authors by Doyle [9], was inspired by a problem encountered by Friedman and Yau about which more will be said later.

8.1. Planar networks. A *planar s-t network* $\mathcal{N}$ is a plane graph with (real) edge lengths, and distinguished vertices s and t on the outside face. The length of a path (from s to t) is the sum of the lengths of its edges.

It is natural to regard two such paths as "adjacent" if one can be obtained from the other by rerouting around a face of $\mathcal{N}$. The paths from s to t then themselves constitute a connected graph. If we place s at the west end and t at the east end of the network, we can identify a southernmost path $\mathcal{P}$ and a northernmost path $\mathcal{Q}$;

a sequence of adjacent paths beginning with $\mathcal{P}$ and ending with $\mathcal{Q}$ will be called a *migration*.

Let us call a migration "b-short" if none of its paths has length greater than b. Doyle asked whether the existence of a migration in which all paths are b-short implies that there is a *strictly northward* b-short migration, that is, one in which each interior face is crossed only once.

It turns out not to be possible to define both a modular lattice and a submodular (length) function in general for this problem, and indeed, somewhat to the authors' surprise, Doyle's question has a negative answer. An example is given in Figure 7; here $b = 40$ and the *only* b-short migration crosses the rightmost interior face three times.

The reader will observe that the edges bd and df are used in both directions by different paths of the migration. If we allow a more liberal view of what a path is, allowing an edge to be used in both directions *in the same "path,"* then we would get around this example. Consider the walk $s\,a\,b\,d\,c\,e\,f\,d\,b\,t$, to be thought of—with a small stretch of the imagination—as obtained from the southernmost path by raising a "spike" $b\,d\,f\,d\,b$, and then rerouting around the upper central face. If this walk is included in the set of paths, then we can take it as the second path in a migration, then reroute around the lower central face, leading to the fifth path in the given migration, and continue as before, getting a strictly northward 40-short migration.

We strongly suspect that, in this more liberal setting, if there is a b-short migration, then there is a strictly northward one. However, we have been unable to prove such a result. To illustrate the difficulties involved in giving the set of "paths" a lattice structure, the reader is invited to consider how best to define the meet of the paths $s\,e\,f\,d\,c\,a\,b\,t$ and $s\,a\,b\,d\,c\,e\,f\,t$, and give that "path" a meaningful length.

One possible solution is to take as our lattice the collection of all subsets S of the set of bounded faces, define $T(S)$ to be closure of the union of the faces in S and the part of the plane below the southernmost path, and define our submodular function by setting $f(S)$ equal to the length of the boundary of $T(S)$. We then seek a migration from the empty set to the whole set. Theorem 6.2 applies immediately: if there is a b-short migration, then there is a "monotone" migration. However, "strictly northward" is no longer a good description. In the example in Figure 7, for instance, the optimum migration starts by taking in the top central face, when the boundary of the "conquered region" S consists of the southernmost path and the boundary of the top face.

Another, arguably more satisfactory, way to obtain a positive result is to require our network to be directed and acyclic, so that each edge is oriented and thus can be traversed in only one direction.

THEOREM 8.2. *Let $\mathcal{N}$ be an acyclic planar s-t network with southernmost path $\mathcal{P}$ and northernmost path $\mathcal{Q}$. Then for any real b, if there is a b-short migration from $\mathcal{P}$ to $\mathcal{Q}$, then there is a b-short migration which crosses each interior face only once.*

Proof. If $\mathcal{A}$ and $\mathcal{B}$ are two s-t paths, put $\mathcal{A} \prec \mathcal{B}$ if they do not cross and $\mathcal{B}$ lies north of $\mathcal{A}$. Since there is a unique way to cross each interior face, this relation induces a sublattice of the lattice of all sets of interior faces, and thus the paths form a distributive lattice. Every arc of $\mathcal{N}$ appears exactly as often in $\mathcal{A}$ and $\mathcal{B}$ combined as it does in $\mathcal{A} \vee \mathcal{B}$ and $\mathcal{A} \wedge \mathcal{B}$ combined; thus the length function is modular and Theorem 6.2 applies. □

We do of course obtain the stronger result that there is a northward migration in $\mathcal{N}$ which lies at or below any other migration in the worm order (applied to the

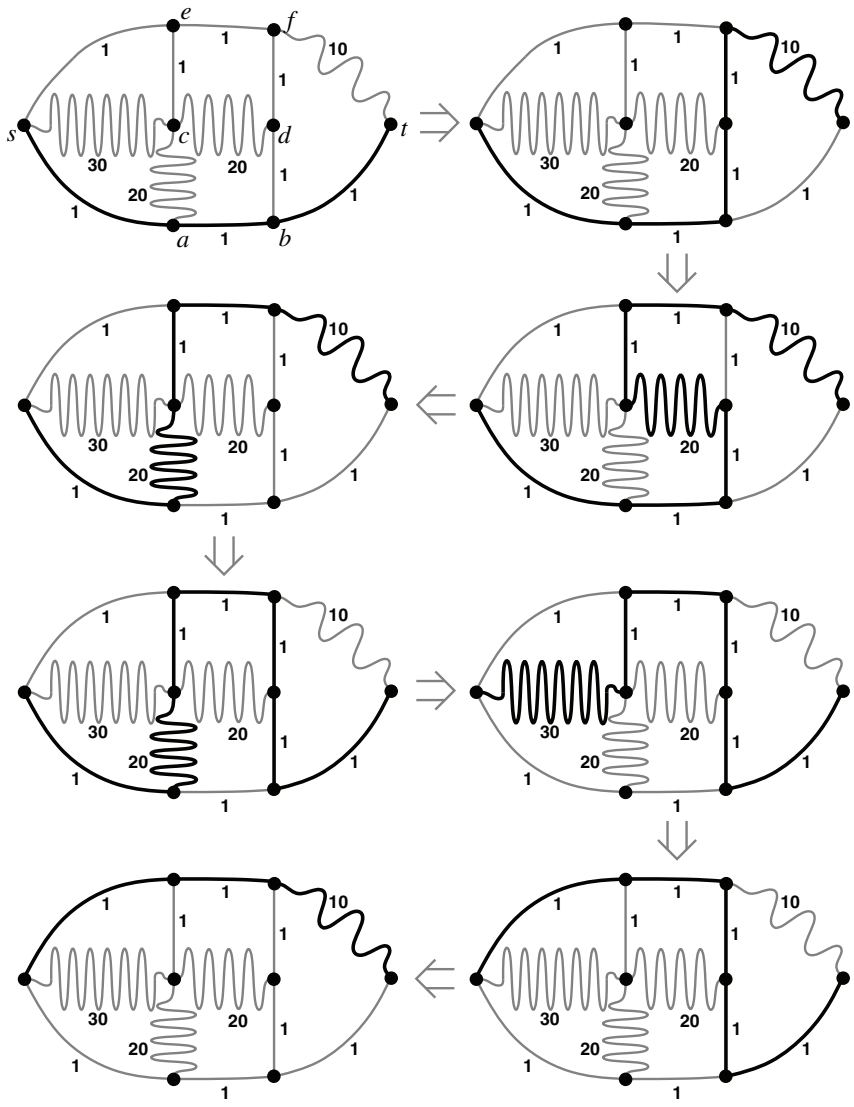

FIG. 7. *A nonstrictly northward unique minimax migration in a planar s-t network.*

sequence of path lengths of the migration). This assures us, for example, that if there is a series of acyclic directed networks (say, from s to t to u etc.), then the problem can be solved separately on each piece and then the migrations combined to obtain one whose maximum total length is minimized.

8.2. Sweep-searching. Continuous versions of Doyle's question, and the other variants in the previous subsection, arise naturally in the following context. We consider searching for a lost child (who may move around) in a field, which we take to be a simply connected region which may have interesting topography. This is often done with a line of searchers who hold hands or maintain some fixed minimum

distance from one another while sweeping the entire field. It is thus natural to ask whether, with a given number of searchers, it might be necessary to retreat, having to cover the same piece of territory twice.

A mathematical version of this problem was encountered by Friedman and Yau in studying the Poincaré Conjecture. Fix two points x and y on the boundary of a disc with Riemannian metric, and let $\mathcal{P}_0$ and $\mathcal{P}_1$ be the arcs of the boundary which they terminate. Suppose there is a homotopy $\mathcal{P}_t : \ t \in [0,1]$ in the space of all continuous arcs from x to y, in which no arc has length exceeding b. Then must there be one in which, for every z in the disc, the set of t for which $z \in \mathcal{P}_t$ is connected?

One might imagine that the disc is a (frozen) glove with metric inherited from $\mathbb{R}^3$, the wrist-hole forming the boundary. A rubber band is stretched along the boundary between two points and pulled over the surface of the glove until it reaches the other side of the wrist-hole, in such a way that the band is never stretched beyond length b. Does it follow that the same can be done, ensuring that each point of the glove is passed over only once?

As far as we know, no proof or counterexample has been found. Our Theorem 6.2 does not appear to apply because, just as in the discrete version considered in the previous subsection, the natural way to define the join of two paths can result; e.g., in a path and a loop.

However, if we do permit this broadening of the notion of path, a continuous version of Theorem 6.2 can be applied. A proof will appear in the authors' forthcoming work [6]. In the context of searching a field, an interpretation of the result is that if the searchers are not required to form one line, and may move freely about the field when not searching, then indeed the field can be swept by a minimal number of searchers in such a way that no spot is multiply swept.

To see that in fact such a capability may reduce the required number of searchers, the reader is invited to imagine a field which contains a very tall, thin "Devil's Tower." If the searchers are required to form a single line, that line must at some stage extend all the way up the tower and down again. In the more permissive setting, however, a group of searchers sufficient to encircle the tower can be sent to the top, walking up without searching. While the remainder of the searchers approach the bottom of the tower, the encirclers descend the tower; when the two groups join, the tower has been miraculously swept, and the augmented line can proceed to search the rest of the field.

8.3. Passing obstacles. The applications in this section can be seen as further variants on the case of a planar directed network. We are still interested in paths from a western point s to an eastern point t, sweeping northwards from the southernmost path to the northernmost; but now we do not require the paths to follow arcs of a given planar network. However, we do impose conditions that, effectively, mean that our paths always proceed from west to east.

Imagine first that there are a number of pegs on a board, of varying heights. We think of these pegs as a sequence $\langle s = (0,0), p_1, \ldots, p_n, t = (1,0)\rangle$ of points in general position in the plane, ordered according to their horizontal coordinate, together with their heights $h_i > 0$. There is a heavy cable attached to the pegs at s and t that starts by running below (i.e., south of) all the pegs. The cable has to be moved until it runs above (north of) all the pegs, lifting it over one peg at a time. We impose a constraint that the cable always runs west-east, i.e., the horizontal coordinate is increasing along the cable, so the position of the cable is given by a continuous function $P : [0,1] \to \mathbb{R}$. If this task can be carried out at all, can it be done by moving the

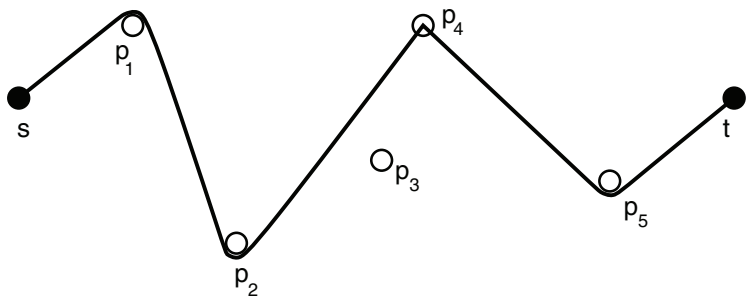

FIG. 8. *A cable routed over p_1, under p_2, over p_3, through p_4, and under p_5.*

cable ever northwards?

Our lattice is $[2]^n$, a product of three-element chains. For each peg p_j, the j-coordinate of a state $S \in [2]^n$ encodes whether the cable passes below the peg ($S_j = 0$), sits on top of the peg ($S_j = 1$), or passes above the peg ($S_j = 2$). We say that a path $P : [0,1] \to \mathbb{R}$ *respects* S if

- whenever $S_j = 0$, P passes below or through p_j,
- whenever $S_j = 1$, P passes through p_j,
- whenever $S_j = 2$, P passes above or through p_j.

See Figure 8. We define $f(S)$ to be the minimum length of a path respecting S, plus the sum of $2h_j$ for all j with $S_j = 1$. Thus f represents the infimum of the length of cable required for state S: the cable is heavy and cannot leave the ground except to pass over a peg.

It is easy to see that f is a submodular function: given states S and T, and shortest paths $P_S, P_T : [0,1] \to \mathbb{R}$, respecting these states, we define paths $P_S \vee P_T$ and $P_S \wedge P_T$ to be the pointwise maximum and minimum of P_S and P_T. Then $P_S \vee P_T$ respects $S \cup T$ and $P_S \wedge P_T$ respects $S \cap T$. The total lengths of the two paths are the same, and $S \vee T$ and $S \wedge T$ collectively require paths to sit on top of the same multiset of pegs as S and T do, so the paths witness $f(S \vee T) + f(S \wedge T) \leq f(S) + f(T)$.

Therefore Theorem 6.2 applies, and we have the result: a path $\mathcal{C}$ in $[2]^n$ with $f(C) \leq L$ for all $C \in \mathcal{C}$ represents a plan to get a cable of length L over all the pegs, and our theorem tells us that such a plan need only involve passing over each peg once.

Essentially the same arguments work if the pegs are replaced by geometric shapes which act as obstacles in the plane, perhaps for a programmed robot.

8.4. Graph searching. We now shift once more to a somewhat different discrete setting, concerning a class of problems introduced by Breisch [5] and Parsons [27]. Thinking of a graph as a network of tunnels, the idea is to find a lost spelunker, or perhaps capture a fugitive, by posting searchers at vertices and moving them from vertex to vertex. We are permitted at each step to remove any number of searchers from the graph, and then to place one server on a vertex or move a server from its vertex to a neighboring vertex, along some edge. An edge is considered "cleared" if a searcher has traversed it, and there has been no subsequent opportunity for the fugitive to reoccupy it; in other words, since the last traversal, there has been no route to it from an uncleared edge that was not blocked by searchers.

It is natural to ask whether, if k searchers suffice to clear the entire graph, then there is a way to do it without allowing any edge to become reoccupied. This would of course imply that any graph can be cleared in linear time with the minimum number of searchers. The question attained considerable notoriety before being answered in the affirmative by LaPaugh [20].

At any time t during a clearing of $G = (V, E)$ by k searchers, the set X_t of cleared edges is of course a member of the distributive lattice of subsets of E, on which the "boundary" function $f(X) := |\{v \in V : \ E(v) \cap X \neq \emptyset \text{ and } E(v) \setminus X \neq \emptyset\}|$, where $E(v)$ is the set of edges incident with vertex v, is easily seen to be submodular. Since the set of cleared edges can grow only one edge at a time, and there must always be searchers at all vertices bounding X_t, we get a slipchain on which f never takes on a value larger than k.

The converse does not quite hold, because it may take an extra searcher to clear an edge. To get around this problem, Bienstock and Seymour [3] tackled a "mixed" version of the problem in which an edge is *also* considered to be cleared when there are searchers at both ends. The mixed problem does match the slipchains and, in effect, this is how they solved it: they used their version of our Theorem 6.1 to show that, in this model, it is never necessary to allow reoccupation of edges. To get a new proof of LaPaugh's result, they reduced to the original version of the problem by subdividing edges.

Megiddo et al. [22] showed that it is NP-complete to decide whether a graph can be cleared by k searchers. Combining this result with the above reduction yields an alternative proof that, for a submodular function f on a distributive lattice $\mathcal{L}$, and a number b, it is NP-complete to determine whether there is a maxchain $\mathcal{C}$ in $\mathcal{L}$ with $f(C) \leq b$ for all C on $\mathcal{C}$.

There is extensive literature on the graph searching problem; see, for instance, the annotated bibliography of Fomin and Thiliakos [10]. Monotonicity results for other versions of the problem can be found in [2, 18, 19, 21, 30], among others. Nonmonotonicity results can be found in, for instance, [11, 32].

Let us now consider a new version of graph searching in which it is the edges, instead of the vertices, which must be blocked. Here it is natural to equip the edges of G with capacities $c : \ E \to \mathbb{R}^+$ and to imagine that at the start each vertex is *contaminated*. At any time we may decontaminate a vertex, but must then immediately redistribute our "blocking material" so that all edges connecting an uncontaminated vertex to a contaminated one are blocked (unless we wish to allow some vertices to be recontaminated). To block edge e requires an amount $c(e)$ of our reusable blocking material. If G can be completely decontaminated without ever using more than b of the blocking material, we say that it is b-decontaminable.

Figure 9 illustrates the 6-decontamination of a triangle with edges of capacity 2, 3, and 4. The graph is shown as a network of tubes which narrow to "cross-sectional area" $c(e)$ in the middle. Black vertices are currently contaminated, while a flashing white node is one undergoing decontamination.

THEOREM 8.3. *If a capacitated graph $G = (V, E, c)$ is b-decontaminable, then there is a b-decontamination sequence (of length $|V|$) which allows no recontamination, and, moreover, has the property that the successive amounts of blocking material in use undercut any other decontamination sequence in the worm order.*

Proof. Here $\mathcal{L}$ is the distributive lattice of all subsets X of V, and f is the cut size, given by $f(X) := \sum_{e \in E:\ |e \cap X| = 1} c(e)$, which is well known (and easy to show) to be submodular. Since $f(X)$ blocking material is necessary and sufficient for X to

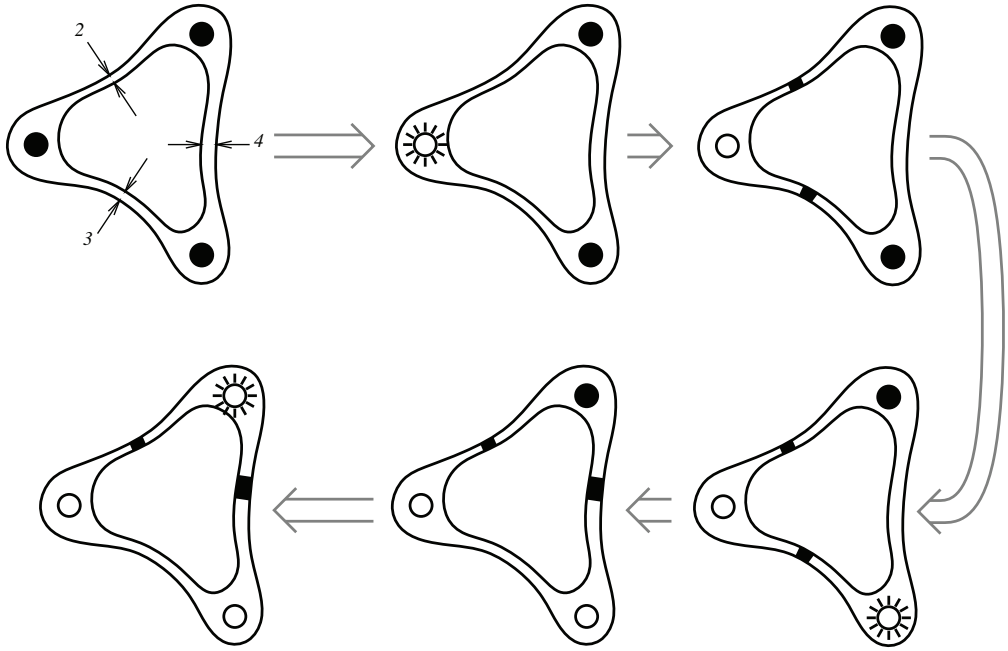

FIG. 9. *Decontaminating a triangle.*

be the set of decontaminated vertices, the slipchain version of Theorem 6.2 completes the proof. □

Theorem 8.3 gives us an attractive way to define a notion of "min-max cut size" $b(N)$ for an s-t network N. Let $\mathcal{L}$ be the lattice of s-t cuts (subsets of V which contain s but not t), and let f be the cut-size as in the previous proof. Define a *migration* to be a sequence of cuts, beginning with $\{s\}$ and ending with $V \setminus \{t\}$, each differing by only one vertex from the previous one. Then we have the following consequence.

COROLLARY 8.4. *In any s-t network N, there is a monotone migration whose cut-sizes undercut, in the worm order, those of any other migration.*

For an s-t network N, we can then define $b(N)$ to be the maximum cut-size among cuts in this optimal migration. Note, however, that our migrations permit augmenting a cut by the addition of a vertex which is not adjacent to any vertex of the cut; in other words, we allow cuts which do not induce connected subgraphs of the network. For *connected* migrations, in which such augmentations are not allowed, the corollary fails. The network of Figure 10, which is the dual of the network in Figure 7, is a counterexample. Its unique optimal monotone migration, with cut sizes, is

$$\{s\}(3),\ \{s,d\}(26),\ \{s,d,b\}(26),\ \{s,d,b,c\}(35),\ \{s,d,b,c,e\}(5),\ \{s,d,b,c,e,a\}(12).$$

We can duplicate this min-max cut size with the connected migration

$$\{s\}(3),\ \{s,a\}(14),\ \{s,a,d\}(35),\ \{s,a,d,b\}(33),\ \{s,d,b\}(26),$$

$$\{s,d,b,c\}(35),\ \{s,d,b,c,e\}(5),\ \{s,d,b,c,e,a\}(12),$$

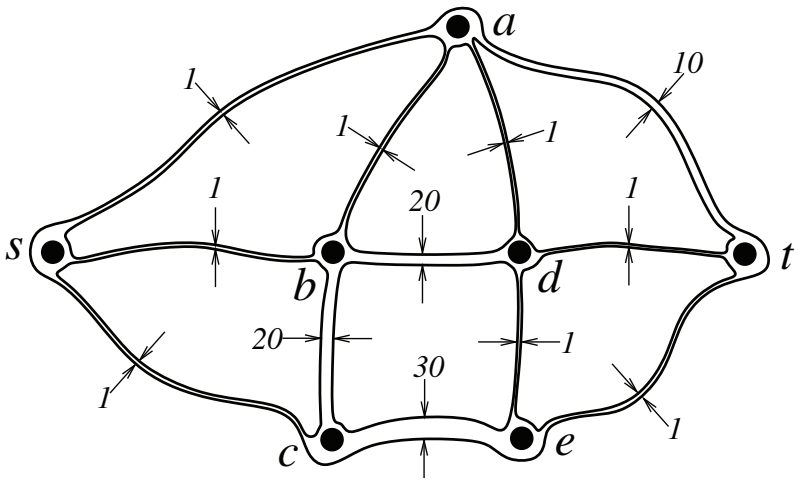

FIG. 10. *An s-t network with no optimal migration which is monotone and connected.*

but the latter requires a backward step—recontamination of vertex a.

This example also illustrates the difference between our edge-cut problems and any version of the original graph-search problems as applied to the line graph. If we consider the vertex-capacitated line graph of the graph in Figure 10, then it is easy to devise a "left-right" search strategy for the line graph using only 35 searchers: there is no need to occupy the vertex *at* of the line graph until all the edges incident with the edge *ce* have been cleared.

8.5. Coordinate percolation. Let us return now to the case of scheduling sequences of real numbers so as to minimize their maximum pairwise sum.

Coordinate percolation takes place in $\mathbb{N}^2$, the positive quadrant of the grid. In contrast to ordinary "independent" site percolation (see, e.g., [4, 16]), the life or death of a site is determined by events associated with the site's two coordinates. In an example of "collision-type" coordinate percolation, each line of the plane grid is assigned a number from the set $\{1, 2, \ldots, k\}$, and a grid point is killed if it gets the same number from its row and column. A northeast walk from the origin represents a way to schedule two $\{1, 2, \ldots, k\}$-sequences so that they never collide, that is, they are never in the same state at the same time. The second author of this paper conjectured in [7] that for large enough k, if the numbers are assigned independently uniformly at random, then, with positive probability, there is an infinite walk on live grid points starting at the origin; this was proven for the unoriented version, where the walk may proceed in any of the four coordinate directions, for $k \geq 4$ in [1, 31].

Gács proved [13] that collision-type coordinate percolation is, in a sense, less tractable than independent percolation; nevertheless, he went on to prove percolation for a variation in [14]. In contrast, the form of coordinate percolation we now consider is proving to be *more* tractable than independent percolation.

In "sum-type" coordinate percolation, real numbers $a_0, a_1, \ldots$ are assigned to the vertical grid lines and numbers $b_0, b_1, \ldots$ to the horizontal grid lines. A site is destroyed if the sum of the numbers assigned to its coordinates exceeds some threshold

t. Note that since the sums constitute a modular function on the grid, Theorem 6.2 assures us that the oriented and unoriented cases coincide here: if there is an infinite walk in the grid starting at the origin, then there is one that proceeds only north and east. Moreover, the point (m, n) is reachable from the origin if and only if the word $\langle a_0, \ldots, a_m \rangle$ precedes $\langle t - b_0, \ldots, t - b_n \rangle$ in the worm order.

Moseman and Winkler [25] (see also [24]) study the case where the a_i and b_j (for $i, j \geq 0$) are drawn independently from the uniform distribution on $[0, 1]$. Letting Θ_t be the probability that one can walk northeast to infinity from the origin on the remaining grid points, it is easy to see that $\Theta_t = 0$ for $t < 1$ and $\Theta_t > 0$ for $t > 1$ since in the former case there are almost surely whole rows and columns that are destroyed, while in the latter the whole x-axis is alive with positive probability.

For the critical point $t = 1$, it is not too hard to see that $\Theta_1 = 0$. In this case, it turns out to be possible to analyze the behavior of Θ_t near the critical point very precisely: Moseman and Winkler give an explicit expression for Θ_t, and show that it is continuous everywhere, C^1 except at the critical point, and has critical exponent $\beta = (3 - \sqrt{5})/2$, i.e.,

$$\frac{\log \Theta_t}{\log(t-1)} \to \beta \text{ as } t \to 1^+.$$

In this critical case, the point (m, n) is reachable from the origin if and only if the word $\langle a_0, \ldots, a_m \rangle$ precedes $\langle 1 - b_0, \ldots, 1 - b_n \rangle$ in the worm order, so the problem is closely related to that of estimating the probability that two random words of given lengths are related in the worm order.

Acknowledgments. We have benefited from conversations with Dana Randall and Robin Thomas of Georgia Tech, and Peter Doyle of Dartmouth College. We would also like to thank two anonymous referees for their very useful comments.

REFERENCES

[1] P. N. Balister, B. Bollobás, and A. M. Stacey, *Dependent percolation in two dimensions*, Probab. Theory Related Fields, 117 (2000), pp. 495–513.
[2] D. Bienstock, N. Robertson, P. D. Seymour, and R. Thomas, *Quickly excluding a forest*, J. Combin. Theory Ser. B, 52 (1991), pp. 274–283.
[3] D. Bienstock and P. D. Seymour, *Monotonicity in graph searching*, J. Algorithms, 12 (1991), pp. 239–245.
[4] B. Bollobás and O. Riordan, *Percolation*, Cambridge University Press, New York, 2006.
[5] R. L. Breisch, *An intuitive approach to speleotopology*, Southwestern Cavers, 6 (1967), pp. 72–78.
[6] G. R. Brightwell and P. Winkler, *Continuous Processes without Backtracking*, in preparation.
[7] D. Coppersmith, P. Tetali, and P. Winkler, *Collisions among random walks on a graph*, SIAM J. Discrete Math., 6 (1993), pp. 363–374.
[8] B. L. Dietrich and A. J. Hoffman, *On greedy algorithms, partially ordered sets, and submodular functions*, IBM J. Res. Develop., 47 (2003), pp. 25–30.
[9] P. Doyle, *private communication*, 2004.
[10] F. V. Fomin and D. Thilikos, *An annotated bibliography on guaranteed graph searching*, available online at http://www.ii.uib.no/∼fomin/fedor/articles1/abgs.pdf.
[11] P. Fraigniaud and N. Nisse, *Monotony properties of connected visible graph searching*, in Proceedings of the 32nd International Workshop on Graphs (WG06), Lecture Notes in Comput. Sci. 4271, Springer, Berlin, 2006, pp. 229–240.
[12] S. Fujishige, *Submodular Functions and Optimization*, Ann. Discrete Math. 47, North–Holland, Amsterdam, 1991.
[13] P. Gács, *The clairvoyant demon has a hard task*, Combin. Probab. Comput., 9 (2000), pp. 421–424.

[14] P. Gács, *Compatible sequences and a slow Winkler percolation*, Combin. Probab. Comput., 13 (2004), pp. 815–856.
[15] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, San Francisco, 1979.
[16] G. Grimmett, *Percolation*, 2nd ed., Springer-Verlag, Berlin, 1999.
[17] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, 2nd ed., Springer-Verlag, Berlin, 1993.
[18] L. M. Kirousis and C. H. Papadimitriou, *Searching and pebbling*, Theoret. Comput. Sci., 47 (1986), pp. 205–218.
[19] L. M. Kirousis and C. H. Papadimitriou, *Interval graphs and searching*, Discrete Math., 55 (1985), pp. 181–184.
[20] A. LaPaugh, *Recontamination does not help to search a graph*, J. Assoc. Comput. Mach., 40 (1993), pp. 224–245.
[21] F. Mazoit and N. Nisse, *Monotonicity of nondeterministic graph searching*, in Proceedings of the 33rd International Workshop on Graphs (WG07), Lecture Notes in Comput. Sci. 4769, Springer, Berlin, 2007, pp. 33–44.
[22] N. Megiddo, S. L. Hakami, M. R. Garey, D. S. Johnson, and C. H. Papadimitriou, *The complexity of searching a graph*, J. Assoc. Comput. Mach., 35 (1988), pp. 18–44.
[23] G. Monge, *Déblai et Remblai*, Mémoires de l'Academie des Sciences, Paris, 1781.
[24] E. Moseman, *The Combinatorics of Coordinate Percolation*, Ph.D. thesis, Dartmouth College, Hanover, NH, 2007.
[25] E. R. Moseman and P. Winkler, *On a form of coordinate percolation*, Combin. Probab. Comput., 17 (2008), pp. 837–845.
[26] N. Narayanan, *Submodular Functions and Electrical Networks*, Ann. Discrete Math. 54, North–Holland, Amsterdam, 1997.
[27] T. D. Parsons, *Pursuit-evasion in a graph*, in Theory and Applications of Graphs, Lecture Notes in Math. 642, Y. Alavi and D. R. Lick, eds., Springer-Verlag, Berlin, 1978, pp. 426–441.
[28] N. Robertson and P. D. Seymour, *Graph minors* X. *Obstructions to tree-decomposition*, J. Combin. Theory Ser. B, 52 (1991), pp. 153–190.
[29] A. Schrijver, *A combinatorial algorithm minimizing submodular functions in strongly polynomial time*, J. Combin. Theory Ser. B, 80 (2000), pp. 346–355.
[30] P. D. Seymour and R. Thomas, *Graph searching and a min-max theorem for tree-width*, J. Combin. Theory Ser. B, 58 (1993), pp. 22–33.
[31] P. Winkler, *Dependent percolation and colliding random walks*, Random Structures Algorithms, 16 (2000), pp. 58–84.
[32] B. Yang, D. Dyer, and B. Alspach, *Sweeping graphs with large clique number*, Discrete Math., to appear.