Realizing Partitions Respecting Full and Partial Order Information

Erik Demaine Jeff Erickson Danny Krizanć Henk Meijer Pat Morin
Mark Overmars Sue Whitesides

Abstract

For $n \in \mathbb{N}$, we consider the problem of partitioning the interval $[0, n)$ into k subintervals of positive integer lengths $\ell_1, \ldots, \ell_k$ such that the lengths satisfy a set of simple constraints of the form $\ell_i \diamond_{ij} \ell_j$ where $\diamond_{ij}$ is one of $<$, $>$, or $=$. In the *full information* case, $\diamond_{ij}$ is given for all $1 \le i, j \le k$. In the *sequential information* information case, $\diamond_{ij}$ is given for all $1 < i < k$ and $j = i \pm 1$. That is, only the relations between the lengths of consecutive intervals are specified. The *cyclic information* case is an extension of the sequential information case in which the relationship $\diamond_{1k}$ between ℓ_1 and ℓ_k is also given. We show that all three versions of the problem can be solved in time polynomial in k and $\log n$.

Keywords: Integer partitions, integer sequences, subset-sum, rhythm pattern, rhythm perception, modular arithmetic

1 Introduction

We consider problems of realizing a sequence having restrictions on its sum and the relative sizes of its terms. In particular, we consider the following problem: Given positive integers n and k, partition $[0, n)$ into k subintervals of positive integer lengths $\ell_1, \ldots, \ell_k$ such that the lengths satisfy a set of simple constraints of the form $\ell_i \diamond_{ij} \ell_j$ where $\diamond_{ij}$ is one of $<$, $>$, or $=$. In the *full information* case, $\diamond_{ij}$ is given for all $1 \le i, j \le k$. In the *sequential information* information case, $\diamond_{ij}$ is given for all $1 \le i \le k$ and $j = i \pm 1$. The *cyclic information* case is an extension of the sequential information case in which the relationship $\diamond_{1k}$ between ℓ_1 and ℓ_k is also given.

For an example of the full information case observe that, for $n = 12$, the comparison matrix in Fig. 1 is satisfied by the sequences $\langle \ell_1, \ldots, \ell_4 \rangle \in \{\langle 1, 1, 8, 2\rangle, \langle 1, 1, 7, 3\rangle, \langle 1, 1, 6, 4\rangle, \langle 2, 2, 5, 3\rangle\}$. On the other hand, for $n = 6$ no solution is possible since the smallest natural sequence satisfying the comparison matrix is $\langle 1, 1, 3, 2\rangle$ and $1 + 1 + 3 + 2 = 7$.

The motivation for studying these types of problems comes from the study of the perception of musical rhythm. Mathematically, a rhythm is a partition of $[0, n)$ into k open intervals called *off-sets* and k integer points called *on-sets* (see References [6, 7, 8, 1, 9]). Musically, we interpret the on-sets as points in time (modulo n) when a percussion instrument is to be

$i \backslash j$	1	2	3	4
1	$=$	$=$	$<$	$<$
2	$=$	$=$	$<$	$<$
3	$>$	$>$	$=$	$>$
4	$>$	$>$	$<$	$=$

Figure 1: A comparison matrix for the full information case.

struck. Experimental evidence shows that humans often do not distinguish between rhythms with the same *rhythmic contour*, i.e. the sequence that specifies whether one off-set is longer than, shorter than or equal to the previous off-set (see References [2, 3, 5, 4]). It then becomes a natural question to ask whether and how a given rhythmic contour can be realized.

In this paper, we give polynomial (in k and $\log n$) time algorithms for all three versions of the problem under study. In particular, for the full information case we give an algorithm that runs in $O(k^2 + \log^c n)$ time, for the sequential information case we give an algorithm that runs in $O(k^4 + \log^c n)$ time, and for the cyclic information case we give an algorithm that runs in $O(k^5 + \log^c n)$ time. The exponent c is given by the time it takes to compute the residue $n \bmod k$ and is not more than 2.

All versions of this problem reduce to special cases of SUBSET-SUM with multiplicity, where there are special constraints on the allowable multiplicities. The efficiency and correctness of our algorithms for solving these problems rely primarily on properties of modular arithmetic. Throughout this paper, we use some standard number-theoretical notations: $\mathbb{Z}_k = \{0, \dots, k-1\}$, $\mathbb{N}_k = \mathbb{Z}_k \setminus \{0\}$, $\mathbb{Z} = \mathbb{Z}_\infty$, $\mathbb{N} = \mathbb{N}_\infty$, and $\mathbb{Z}_k^+$ is the group whose elements are $\mathbb{Z}_k$ and whose operator is addition modulo k.

The remainder of the paper is organized as follows. In Section 2 we given an algorithm for the full information case. In Section 3 we given an algorithm for the sequential information case. In Section 4 we give an algorithm for the cyclic information case. Finally, Section 5 summarizes our results and concludes with directions for future research.

2 Full Information

In this section we consider the full information case in which n and k are given and, for each $1 \le i, j \le k$ we are told either that $\ell_i < \ell_j$, $\ell_i > \ell_j$ or $\ell_i = \ell_j$. We assume that this information is given (implicitly or explicitly) in the form of a comparison matrix $\diamond$ so that we can determine in constant time which of the three cases applies to ℓ_i and ℓ_j. The algorithm we describe will either find a sequence $\ell_1, \dots, \ell_k \in \mathbb{N}$ such that $\sum_{i=1}^{k} \ell_i = n$ and $\ell_i \diamond_{ij} \ell_j$ for all $1 \le i, j \le k$ or the algorithm will conclude that no such sequence exists.

We first observe that, because we are given the entire comparison matrix $\diamond$, we can run any reasonable sorting algorithm to partition $1, \dots, k$ into $m \le k$ equivalence classes $C_1, \dots, C_m$ where (1) $\ell_i = \ell_j$ if i and j belong to the same class and (2) $\ell_i < \ell_j$ if $i \in C_{i'}$, $j \in C_{j'}$ and $i' < j'$.

Refer to Fig. 2. Now our problem is to find $v_1, \dots, v_m \in \mathbb{N}$ such that $v_i < v_{i+1}$, for all $1 \le i < m$ and $\sum_{i=1}^{m} v_i |C_i| = n$. Then by assigning $l_i = v_{i'}$ for all $i \in C_{i'}$ we obtain a solution to the original problem. The restriction $v_i < v_{i+1}$ is slightly inconvenient and we can remove it with a rewording of the problem. Let $t_1 = k$, and let $t_i = t_{i-1} - |C_{i-1}|$ for $1 \le i \le m$. Then it suffices to find $w_1, \dots, w_m \in \mathbb{N}$ such that

$$\sum_{i=1}^{m} w_i t_i = n \quad . \tag{1}$$

From $w_1, \dots, w_m$ we can compute the value of v_i as $v_i = \sum_{j=1}^{i} w_j$. That is, each value w_i represents the increase from v_{i-1} to v_i.

At this point it is tempting to apply dynamic programming immediately to solve (1) directly. However, this would lead to an algorithm with running time $O(k^a n^b)$, for some constants a and b. In general, this is superpolynomial in the input size since the input is a $k \times k$ comparison matrix and an integer n, all of which can be encoded in $O(k^2 + \log n)$ bits. In the following, we describe a representation that allows us to reduce the dependence on n.

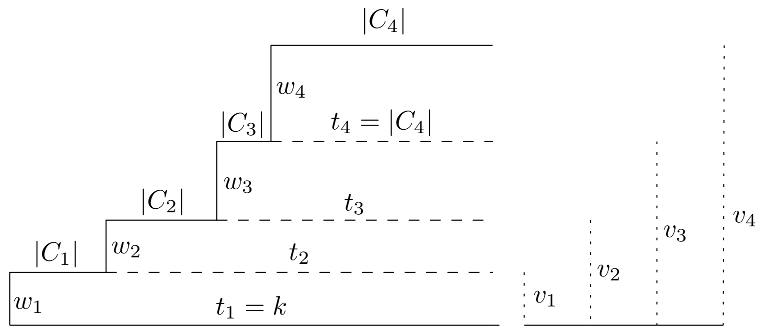

Figure 2: The relationship between $v_1, \dots, v_k$ and $w_1, \dots, w_k$. The area under the curve is n.

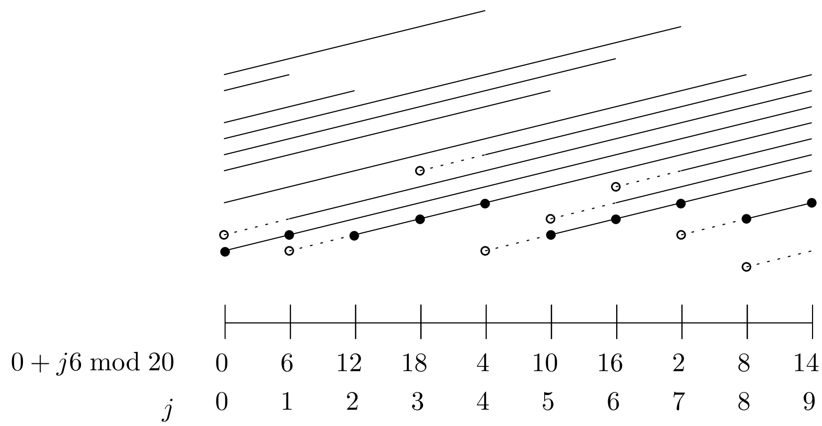

Figure 3: A possible lower envelope used for the set q_0, with $k = 20$ and $\ell_i = 6$. The empty circles show values in D_{i-1} and the filled disks show values in D_i.

Let

$$S_i = \left\{ \sum_{j=1}^{i} w_j t_j : w_1, \dots, w_i \in \mathbb{N} \right\} .$$

Our algorithm uses dynamic programming to compute S_i for $i = 1, \dots, m$. However, since the set S_i has infinite size, we require a compact representation for it. To obtain a nice representation, we observe that, because $t_1 = k$, if S_i contains x then S_i also contains $x + k$, $x + 2k$, $x + 3k$, and so on. Thus, we can represent S_i by storing, for each $y \in \mathbb{Z}_k$, the value

$$D_i(y) = \min(\{\infty\} \cup \{x : x \equiv y \pmod{k} \text{ and } x \in S_i\}) .$$

Lemma 1. *Given D_{i-1}, D_i can be computed in $O(k)$ time.*

Proof. We show that a careful reordering of the elements of $\mathbb{Z}_k$ allows us to compute D_i by a sequence of k/r lower envelope computations each taking time $O(r)$; here, $r = k/\gcd(k, \ell_i)$ is the length of the *orbit* of ℓ_i in the group $\mathbb{Z}_k^+$. The example of the lower envelope in Fig. 3 may be useful in what follows.

Define $q_{c,j} = (c + jt_i) \bmod k$. We will show how to compute $D_i(y)$ for all y in

$$q_0 = \{q_{0,1}, q_{0,2}, q_{0,3}, \dots, q_{0,r}\}$$

in $O(r)$ time. The same algorithm can be used for the sets $q_1, q_2, \ldots, q_{k/r}$ to give a total running time of $O(r) \times k/r = O(k)$. The main observation we use is that

$$f(j) = D_i(q_{0,j}) = \min\{D_{i-1}(q_{0,(j-x) \bmod k}) + x\ell_i : x \in \mathbb{N}_k\} \ . \tag{2}$$

That is, the univariate function $f(j)$ is the lower envelope of k half-lines, where the xth half-line is given by the equation $y = (D_{i-1}(q_{0,x}) + (j-x)\ell_i) \bmod k$, $j \geq 1$. Since the slope, ℓ_i, of all k half-lines is identical and positive and their left endpoints are sorted (by j) the lower envelope can easily be computed in $O(r)$ time by scanning from left to right and keeping track of the current minimum line. This completes the proof. □

Note that the algorithm implied by Lemma 1 is actually very simple, and is given by the following pseudocode

$r \leftarrow \gcd(k, \ell_i)$
for $c = 0, \ldots, k/r - 1$ **do**
  $m \leftarrow \infty$
  for $j = 1, \ldots, r$ **do**
    $m \leftarrow \min\{m, D_{i-1}((c - j\ell_i) \bmod k) + j\ell_i\}$
  end for
  for $j = 0, \ldots, r-1$ **do**
    $D_i((c + j\ell_i) \bmod k) \leftarrow m$
    $m \leftarrow \min\{m, D_{i-1}((c + j\ell_i) \bmod k)\} + \ell_i$
  end for
end for

Once we have computed D_m, we can test if n is in the set S_m by checking if $D_m(n \bmod k) \leq n$. This completes the proof of our first result:

Theorem 1. *The realization problem with full information can be solved in $O(k^2 + \log^c n)$ time.*

3 Sequential Information

Next we consider the realization problem given only sequential information. That is, for each $i \in \{1, \ldots, k-1\}$ we are told only that $\ell_i > \ell_{i+1}$, $\ell_i < \ell_{i+1}$ or $\ell_i = \ell_{i+1}$. Our approach is similar to that of the full information case. By scanning for $\diamond_{i,i+1}$ for $i = 1, \ldots, k-1$ we determine a set of $m \leq k$ equivalences classes $C_1, \ldots, C_m$ over $1, \ldots, k$ such that (1) $l_i = l_j$ if i and j belong to the same class and (2) if $i \in C_{i'}$ and $j \in C_{i'+1}$ then either $l_i < l_j$ or $l_i > l_j$, as indicated by $\diamond$.

Let $t_1 = k$, and let $t_i = t_{i-1} - |C_{i-1}|$ for $i > 1$. Let $s_1 = +1$ and, for $i > 1$, let $s_i = +1$ if the elements in C_i should be greater than the elements in C_{i-1} and let $s_i = -1$ if the elements in C_i should be less than the elements in C_{i-1}. Then, our problem is to find $w_1, \ldots, w_m \in \mathbb{N}$ such that

$$\sum_{j=1}^{m} w_j s_j t_j = n \tag{3}$$

and

$$\sum_{j=1}^{i} w_j s_j \geq 1 \text{ for all } i \in \{1, \ldots, m\}. \tag{4}$$

We say that $w_1, \ldots, w_m$ are *admissible* if they satisfy (4).

Given $w_1, \ldots, w_m$ satisfying (3) and (4), we can compute the value of $l_i \in C_{i'}$ as $l_i = \sum_{j=1}^{i'} w_j s_j$. That is, the value w_j represents the difference in the values in C_{j-1} and C_j, this difference being an increase if $s_j = +1$ and a decrease if $s_j = -1$.

As before, because $t_1 = k$ and $s_1 = +1$, we can implicitly represent the set

$$S_i = \left\{ \sum_{j=1}^{i} w_j s_j t_j : w_1, \ldots, w_i \in \mathbb{N} \text{ and } w_1, \ldots, w_i \text{ are admissible} \right\}$$

by maintaining, for each $y \in \mathbb{Z}_k$ the value

$$D_i(y) = \min \{x : x \in S_i \text{ and } x \equiv y \pmod{k}\} \ .$$

However, unlike the case for full information, the function D_{i-1} is not sufficient for computing the function D_i. In particular, which values of w_i are admissible depends on $\sum_{j=1}^{i-1} w_j s_j$, which can be different for each value of y. Instead, we maintain a two-dimensional table

$$D_i(y,h) = \min \left(\{\infty\} \cup \left\{ x \equiv y \pmod{k} : \begin{array}{l} \text{there exists admissible } w_1, \ldots, w_i \text{ such that} \\ \sum_{j=1}^{i} w_j s_j t_j = x \text{ and} \\ \sum_{j=1}^{i} w_j s_j = h \end{array} \right\} \right) \ .$$

Next we consider exactly how much information must be stored in order to maintain the table D_i. Since $y \in \mathbb{Z}_k$ we know that the first dimension (y) of the table is of size k. The following lemma shows that the second dimension (h) is also not too big.

Lemma 2. *Let $H = k^2 + 1$. If $h \geq H$ then $D_i(y,h) - kt_i \geq D_i(y, h-k)$.*

Proof. Let $w_1, \ldots, w_i$ be any admissible sequence that defines $D_i(y,h)$. That is, $\sum_{j=1}^{i} w_j s_j = h$ and $\sum_{j=1}^{i} w_j s_j t_j = D_i(y,h)$. Let $i' \leq i$ be the largest index such that $w_{i'} \geq k+1$ and $s_{i'} = +1$. The existence of i' is guaranteed by the pigeonhole principle and the assumption that $h > H$. Consider the sequence $w'_1, \ldots, w'_i$ where

$$w'_j = \begin{cases} w_j - k & \text{if } j = i' \\ w_j & \text{otherwise} \end{cases}$$

Then

$$\sum_{j=1}^{i} w'_j s_j = \sum_{j=1}^{i} w_j s_j - k = h - k$$

and

$$\sum_{j=1}^{i} w'_j s_j t_j = \sum_{j=1}^{i} w_j s_j t_j - kt_{i'} \leq \sum_{j=1}^{i} w_j s_j t_j - kt_i \ .$$

Thus, $D_i(y, h-k) < D_i(y,h) - kt_i$ provided that $w'_1, \ldots, w'_i$ is admissible. To see that $w'_1, \ldots, w'_i$ is admissible we observe that, if this were not the case, then there must exist some index $r > i'$ such that $\sum_{j=1}^{r} w_j s_j \leq k$. But then, by the pigeonhole principle there must exist some index $i'' > r > i'$ such that $w_{i''} > k$ and $s_{i''} = +1$. But this is not possible since i' was chosen to be the largest index with this property. $\square$

Lemma 2 shows that in computing D_m we need only consider values of $h \leq H$. This is because for any value x that appears as $x = D_m(y,h)$ for $h > H$, there is a value $z < x$ that appears as $z = D_m(y,h')$ with $h' \leq H$. Since $D_m(y,h')$ implicitly represents the set $\{z, z+k, z+2k, \ldots\}$, the value x is represented by $D_m(y,h')$.

Thus, to obtain our final answer, we need only compute a table D_m containing Hk entries. However, a small technicality occurs because computing D_m from D_{m-1} requires (as we shall see) looking up table entries of the form $D_{m-1}(y,h)$ where $H < h < H+k$. The easiest way to deal with this is to use a table of size $(H+k)k$ to store D_{m-1}. But then to compute D_{m-1} from D_{m-2} we require table entries of the form $D_{m-2}(y,h)$ where $H < h < H+2k$, and so on. In general, the table D_i will have $(H+k(m-i))k$ entries so that we can lookup any value $D_i(y,h)$ with $y \in Z_k$ and $1 \le h \le H+(m-i)k$. Note that this only increases the sizes of the tables by a constant factor, and the following lemma shows that we can compute these tables in time proportional to their size.

Lemma 3. *Given D_{i-1}, D_i can be constructed in $O(Hk)$ time.*

Proof. We first describe the algorithm for the case $s_i = +1$. The algorithm for the case $s_i = -1$ is similar except for a small modification described at the end of the proof.

As in the proof of Lemma 1 we reduce the problem to a sequence of lower-envelope computations. As before, we begin by splitting the elements of $\mathbb{Z}_k$ into the sets $q_0, \ldots, q_{k/r}$ where $r = \gcd(k, t_i)$ and $q_{c,j} = (c + jt_i) \bmod k$. Using exactly the same scanning algorithm used in Lemma 1 we can compute $D_i(q_{0,j}, j)$ for all $1 \le j \le H+(m-i)k$ in $O(H)$ time. Again, this is because the univariate function

$$f(j) = D_i(q_{0,j}, j) = \min\{D_{i-1}(q_{0,j-x}, j-x) + xt_i : 1 \le x \le j\}$$

is the lower envelope of $H+(m-i)k$ parallel half-lines. By repeated applications of the above procedure we can compute $D_i(q_{0,j}, j+c)$ for all $1 \le j \le H$ and all $0 \le c < r$ in $O(Hr)$ time. Finally, by repeating that procedure k/r times we compute entire table $D_i(y,h)$, for all $y \in Z_k$ and all $1 \le h \le H$ in $O(Hk)$ time, as required.

The case $s_i = -1$ is handled in a symmetric manner except that now the function f is defined as

$$f(j) = D_i(q_{0,j}, j) = \min\{D_{i-1}(q_{0,j+x}, j+x) - xt_i : 1 \le x \le \infty\} \ .$$

The difficulty with this formulation is that $f(j)$ is the lower envelope of an infinite number of lines. However, it follows immediately from Lemma 2 that

$$f(j) = D_i(q_{0,j}, j) = \min\{D_{i-1}(q_{0,j+x}, j+x) - xt_i : 1 \le x \le H - j + (m-i+1)k\} \ .$$

Thus, we can compute D_i by taking the lower envelope of $H+(m-i+1)k$ parallel half-lines. This completes the proof. □

We have just shown that we can incrementally construct the sets $S_1, \ldots, S_m$ in $O(Hk) = O(k^3)$ time per set. This yields our second theorem:

Theorem 2. *The realization problem with sequential information can be solved in $O(k^4 + \log^c n)$ time.*

4 The Cyclic Information Case

In this section we consider the cyclic version of the realization problem. The cyclic version is identical to the sequential version except that one additional constraint, namely the relationship between ℓ_1 and ℓ_k, is given. We show that the cyclic version of the problem can be solved using $O(k)$ applications of the algorithm for the sequential version of the problem.

Let $t_1, \ldots, t_m$ and $s_1, \ldots, s_m$ be defined as in the previous section and suppose that there exists $w_1, \ldots, w_m \in \mathbb{N}$ such that

$$\sum_{j=1}^{m} w_j s_j t_j = n \ ,$$

$$\sum_{j=1}^{i} w_j s_j \geq 1 \text{ for all } i \in \{1, \ldots, m\} \ ,$$

$$w_1 s_1 \leq \sum_{j=1}^{i} w_j s_j \text{ for all } i \in \{3, \ldots m-1\} \ ,$$

and

$$w_1 s_1 < \sum_{j=1}^{i} w_j s_j \text{ for all } i \in \{2, m\} \ .$$

Then, rearranging the above equations we get the equivalent statements

$$\sum_{j=2}^{m} w_j s_j t_j = n - w_1 t_1 = n - w_1 k \tag{5}$$

and

$$\sum_{j=2}^{i} w_j s_j \geq 0 \text{ for all } i \in \{3, \ldots, m-1\} \ . \tag{6}$$

$$\sum_{j=2}^{i} w_j s_j \geq 1 \text{ for all } i \in \{2, m\} \ . \tag{7}$$

Note that Equations (5)–(7) are almost identical to Equations (3) and (4) and that the existence of $w_2, \ldots, w_m$ satisfying these equations can be tested in $O(k^4 + \log^c n)$ time using the algorithm from the previous section. This means that, if there exists a solution to our cyclic information problem in which the elements of class C_1 are assigned a value not exceeding any value assigned to any other class C_i, $i \neq 1$, then we can find this solution in $O(k^4 + \log^c n)$ time. However, if there exists any solution then at least one of the classes C_i must be assigned a minimum value in this solution. Thus, by running the algorithm from the previous section m times we can determine if there exists any solution.

Theorem 3. *The realization problem with cyclic information can be solved in $O(k^5 + \log^c n)$ time.*

5 Conclusions

We have considered the problem of partitioning the interval $[0, n)$ into k positive integer length subintervals satisfying some simple order requirements. The types of requirements we have considered include full information, in which the relative length of each pair of subintervals is given, sequential information, in which only the relative lengths of consecutive subintervals is given, and cyclic information in which the relationships between consecutive subintervals and the first and last subinterval are given. Our algorithms run in $O(k^2 + \log^c n)$, $O(k^4 + \log^c n)$ and $O(k^5 + \log^c n)$ time, respectively.

The most general version of this class of problems is as follows: Given any subset of the order matrix $\diamond$, find a sequence $l_1, \ldots, l_k \in \mathbb{N}$ that respects all relations in this matrix and whose sum is n. This remains an open problem.

Another problem, whose solution would be useful in performing perceptual tests on rhythms, is that of selecting uniformly at random from all partitions of $[0, n)$ that satisfy some sequential, cyclic or total information constraints. Such an algorithm would be useful for testing hypotheses of the form: "All rhythms of length n and having k onsets that satisfy some set of constraints sound alike to most listeners."

The sequential and cyclic information problems we study are motivated by the 3-level $(+-0)$ contour representation studied by Dowling [2]. This representation has been generalized to multi-level contours [5, 4] where we are given, for each l_i, a range relative to l_{i-1}. For example, we may be told that $l_i \in [l_{i-1}+50, l_{i-1}+100]$. The problem is then to find $l_1, \ldots, l_k$ that satisfy all these constraints and whose sum is n.

Acknowledgements

This work was initiated at the Bellairs Winter Workshop on Computational Geometry for Music Information Retrieval, January 28 to February 4th, 2005. The authors are grateful to Godfried Toussaint for organizing the workshop and presenting the open problems which lead to the current paper. The authors are also grateful to the other workshop participants, namely Greg Aloupis, David Bremner, Justin Colannino, Mirela Damian, Vida Dujmović, Francisco Gomez, Ferran Hurtado, John Iacono, Stefan Langerman, Erin McLeish, Suneeta Ramaswami, David Rappaport, Diane Souvaine, Ileana Streinu, Perouz Taslakian, Remco Veltcamp, and David Wood, for providing a stimulating work environment.

References

[1] M. Diaz-Banez, G. Farigu, F. Gomez, D. Rappaport, and G. T. Toussaint. El compas flamenco: A phylogenetic analysis. In *Proceedings of BRIDGES: Mathematical Connections in Art, Music, and Science*, pages 61–70. Southwestern College, 2004.

[2] W. J. Dowling. Scale and contour: Two components of a theory of memory for melodies. *Psychological Review*, 85(4):341–354, 1978.

[3] T. Fujioka, L. J. Trainor, B. Ross, R. Kakigi, and C. Pantev. Musical training enhances automatic encoding of melodic contour and interval structure. *Journal of Cognitive Neuroscience*, 2004.

[4] Y. E. Kim, W. Chai, R. Garcia, and B. Vercoe. Analysis of a contour-based representation for melody. In *Proceedings of International Symposium on Music Information Retrieval*, 2000.

[5] Adam T. Lindsay. Using contour as a mid-level representation of melody. Master's thesis, MIT Media Lab, 1996.

[6] G. T. Toussaint. A mathematical analysis of African, Brazilian, and Cuban clave rhythms. In *Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science*, pages 157–168. Townson University, 2002.

[7] G. T. Toussaint. Algorithmic, geometric, and combinatorial problems in computational music theory. In *Proceedings of X Encuentros de Geometria Computacional*, pages 101–107. University of Sevilla, 2003.

[8] G. T. Toussaint. Classification and phylogenetic analysis of African ternary rhythm timelines. In *Proceedings of BRIDGES: Mathematical Connections in Art, Music, and Science*, pages 25–36. University of Granada, 2003.

[9] G. T. Toussaint. Computational geometric aspects of musical rhythm. In *Abstracts of the 14th Annual Fall Workshop on Computational Geometry*, pages 47–48. Massachussetts Institute of Technology, 2004.