A NEW ALGORITHM FOR
HORIZONTAL SPACING OF PRINTED MUSIC

Lippold Haken
CERL Sound Group, University of Illinois
L-Haken@uiuc.edu

Dorothy Blostein
Department of Computer and Information Science
Queen's University, Kingston, Ontario

ABSTRACT: Much has been published on text processing algorithms and text justification. The analogous problem in music notation, the horizontal spacing of notes, is much more complex and there are no standard algorithms. For text justification, the most famous algorithm (developed by Knuth) is based on a model of boxes and springs. Each word is in a box, and all the boxes are connected to neighboring boxes with springs. The line of text is justified by applying a force at each end of the line, and stretching the springs until the line is the proper length. This algorithm is not immediately applicable to music, because note placement must emphasize temporal relationships, and music often has parallel staves, each with their own symbol widths. We present a modified Knuth algorithm used in the Lime music notation editor.

INTRODUCTION: The aesthetic requirements of proper music spacing are difficult to quantify and automate. In this paper we describe a new algorithm used by the Lime music notation editor. Lime's music representation specifies pitch and duration information for each note (Haken and Blostein, 1993); from this information we wish to produce aesthetically pleasing printed music. Lime allows the user to explicitly modify note spacing in a variety of ways, but the goal is to minimize the need for such human intervention.

This work was motivated by comments from Lime users. It has similarities to two existing algorithms (Blostein and Haken, 1991) (Gourlay, 1987) but differs significantly by its use of duration-dependent spring constants. The algorithm proceeds in two steps. First, each system in the piece of music is inspected to create a list of springs and a list of rods. The springs are the musical counterpart to the springs (or "glue") Knuth uses in text processing (Knuth and Plaas, 1981). The rods span some subsection of a system and have a length determined by the width of symbols which they span. In the second step of the algorithm, the springs are stretched to match or exceed the lengths of the rods, and thus the final spacing of notes is determined.

SPRINGS: In the figure below, vertical dashed lines mark the musical simultaneities. This piece has parallel staves; at every simultaneity one or more of the staves has a new note, chord, or rest. One spring is created between each successive simultaneity, as shown in the figure. The spring constant for each spring is derived from the duration of the shortest note sounding at the time of the spring. This shortest note may have started at the simultaneity or it may be held over from a previous simultaneity. The function relating duration to spring constant is inverse logarithmic, with parameters controllable by the user. Longer durations correspond to smaller spring constants (the spring stretches more easily); shorter durations correspond to larger spring constants (more force is to be applied to stretch these springs).

Often musical symbols (such as clef changes) appear between notes. The symbols generally do not have springs associated with them. One exception is the bar line; if all staves in the system have a bar at the same time (the usual case for bar lines,) a spring with a very large spring constant is created to represent the bar line's white space.

RODS: Rods are created between successive notes, chords, or rests on each staff. Each staff is treated in isolation. The length of the rod corresponds to the total width of the noteheads, accidentals, flags, dots, lyrics, clefs, or other symbols which appear between the notes on the staff. Since the staves are treated in isolation and one staff may have longer notes than another, rods can span several simultaneities. Thus rods can span several springs, and they can span (or partially span) rods from parallel staves.

In addition to the rods described above, the algorithm creates a rod which spans the whole length of the staff. Also, if the user explicitly specifies the length of "neighborhoods" in the music, rods which span the length of each neighborhood are created. The horizontal bars in the figure represent rods.

STRETCH THE SPRINGS: The second step of the algorithm is the stretching of the springs. Each rod spans one or more springs. These springs must be stretched to the length of the rod. The amount of force required to stretch the springs to the length of the rod depends on the spring constants. If the rod spans several springs with a variety of spring constants, the same amount of force stretches springs with the smaller spring constants more than those springs with larger constants.

Since rods can span other rods, a longer rod may span springs which are already stretched due to a shorter rod. The shorter rod exerts a certain amount of force on the springs it spans; those springs are only
stretched by the longer rod if the longer rod exerts an even greater force. In some cases shorter rods exert so much force on the springs that the longer rod need not exert any additional force; the shorter rods may already stretch the springs to equal or exceed the length of the longer rod. Similarly, a longer rod may exert so much force that the springs spanned by a shorter rod are already stretched to equal or exceed the length of the shorter rod.

The spring stretching step begins by considering all the rods which span a single spring. After each spring is stretched to the length of the corresponding rod, the rod is discarded and the force applied to the spring is stored with the spring.

Next, the amount of force which each remaining rod must exert on the springs it spans is computed. This force may stretch a subset of the springs spanned by the rod; some of the springs may already be stretched by a greater force.

The rod which must exert the maximal force is chosen from the remaining rods. This force is applied to all the springs the rod spans which do not already have a greater force applied to them, and the rod is discarded. If the rod spanned or partially spanned any of the remaining rods, the amount of force those rods must exert is recomputed. This sequence of processing and discarding the rod with the maximal force is repeated until no rods remain.

A musical excerpt after springs have been stretched. Vertical dashed lines mark simultaneities. Rods are indicated by heavy horizontal lines; rod length is the length of the line, rod span is up to the next rod.

FINALLY: When all the rods have been processed and discarded, the stretched springs indicate the final spacing of the music. The notes fill out the width of the staves because the algorithm includes a rod which spans the width of the staves. In extreme situations the music may actually exceed the staff width; this can only occur if the user asked to put so many notes on a line that the symbols overlap. In this situation it is not possible to print aesthetically pleasing music, but Lime attempts to minimize the impact of overlapping symbols. The music is linearly compressed to the length of the staff; this compression not only moves notes closer together, but other symbols, such as accidentals, move closer to the notes.


ICMC PROCEEDINGS 1995