PLOrk: The Princeton Laptop Orchestra, Year 1

Daniel Trueman[*], Perry Cook[†], Scott Smallwood[*], and Ge Wang[†]

[*]Music Department, Princeton University
(dan|skot)@music.princeton.edu
[†]Computer Science Department, Princeton University
(prc|gewang)@cs.princeton.edu

Abstract

In this paper we report on the current state of the newly established Princeton Laptop Orchestra (PLOrk), a collection of 15 meta-instruments each consisting of a laptop computer, interfacing equipment, and a hemispherical speaker. Founded in the fall of 2005, PLOrk represents the first laptop ensemble of its size and kind, and brings together many of our research and aesthetic interests as musicians, composers, and computer scientists. Here we chronicle the first steps of the ensemble, including details about the technology, the music, compositional challenges, and what we have learned in the process.

1 Introduction

The Princeton Laptop Orchestra (PLOrk) is a newly established ensemble of computer-based musical meta-instruments (see Bahn and Trueman, 2001, and Bahn, Hahn, and Trueman, 2001, for more about meta-instruments). Each instrument consists of a laptop, a multi-channel hemispherical speaker, a variety of control devices (key-boards, graphics tablets, sensors, and others), and software developed in the ChucK and Max/MSP languages/environments. The students who make up the ensemble act as performers, researchers, composers, and software developers. The challenges are many: what kinds of sounds can we create? how can we physically control these sounds? how do we compose with these sounds? There are also social questions with musical and technical ramifications: how do we organize a fifteen players in this context? with a conductor? via a wireless network? The ensemble represents a culmination of research and practice in the areas of live computer music performance, group improvisation, spatialization, the physical modeling of instruments and their patterns of sound radiation, computer music programming languages and real-time performance, and computer music pedagogy.

This paper represents our first public documentation since the establishment of PLOrk in the fall of 2005. Here we will describe the historical backdrop of PLOrk, its origins and motivations; followed by a brief technical description of the PLOrk meta-instrument and its development, a look at group organization, networking and communication, and spatialization; and finally, we will discuss some of the ongoing problems we have encountered, and the compositional challenges PLOrk sets forth.

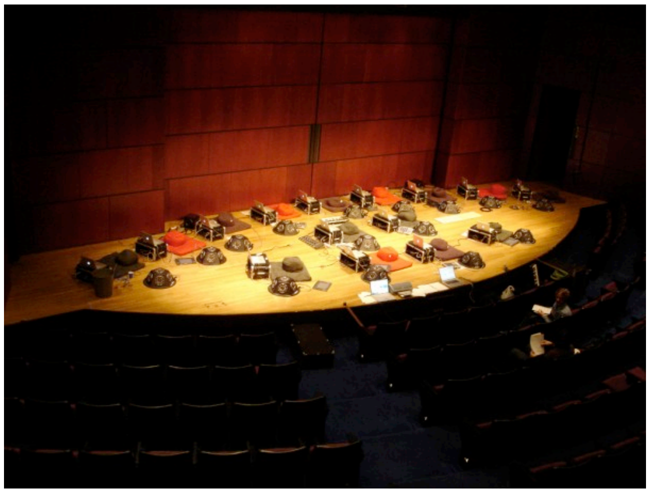
Figure 1.1. PLOrk (without players) at its first concert

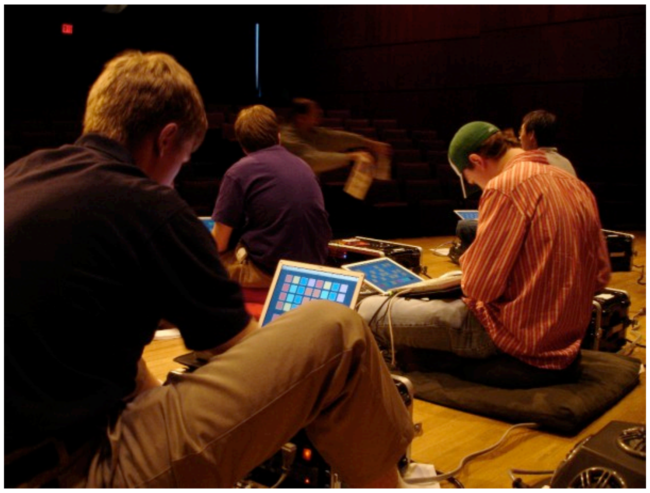
Figure 1.2. A view from within PLOrk

1.1 Motivations and Predecessors

The motivation for PLOrk emerged from several bodies of research which include designing spherical speakers that have a more instrument-like presence (Cook and Trueman 1998; Wessel 1991; Trueman, Bahn, Cook 2000), human-computer interface designs that involve perfomers physically the way musical instruments do (Trueman and Cook 2000), software to link the performers's body to sound (Cook and Scavone 1999, Trueman and DuBois 1997), computer music programming language design for composition and performance (Wang and Cook 2003, Wang and Cook 2004). In the past, we have explored these ideas with small groups of people (2-3) (Bahn and Trueman 2001), and in the fall of 2005 we initiated the Princeton Laptop Orchestra to extend these ideas to larger groups, using the orchestra (in a very general sense) as a model. We are also inspired by the work of other technologically based ensembles, including *The Hub* (http://en.wikipedia.org/wiki/The_Hub_(band)) and *Mimio* (*Music in Movement Electronic Orchestra*, http://www.gjp.info/infomimeo.htm).

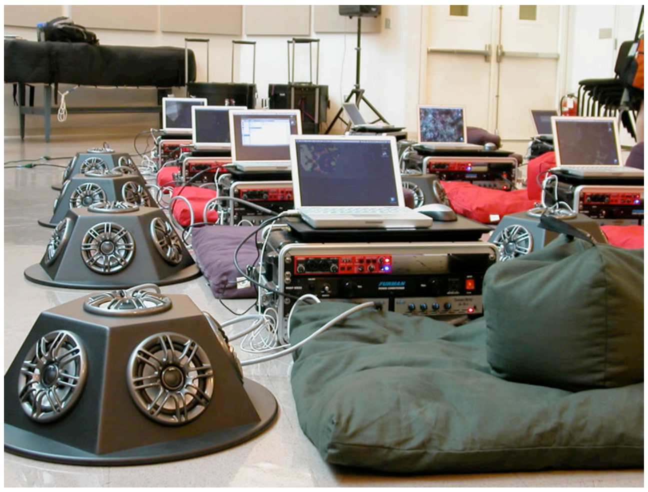
Figure 1.3. PLOrk in rehearsal

2 PLOrk Dissected

2.1 Anatomy of a PLOrk Meta-Instrument

Each meta-instrument in PLOrk consists of the following:

- A laptop computer, currently an Apple 12-inch 1.5 GHz PowerBook G4, utilizing the software development environments of Max/MSP, SuperCollider, and ChucK.
- A rack of audio equipment consisting of a multi-channel firewire interface (currently an Edirol FA-101), speaker amplification (currently a Stewart DA-70-2 2-channel amp, and a Stewart DA-70-4 4-channel amp), and a sensor interface (ElectroTap Teabox)
- A hemispherical speaker with six individually addressable speakers.

In addition to the above, we have a collection of interfacing devices and sensors that can be integrated into any of the meta-instruments to provide physical control of expression. These include off-the-shelf keyboards, percussion pads, and knob/slider controllers, but also custom interfaces using sensors such as accelerometers, pressure pads (using force-sensing-resistors), proximity sensors, light sensors, etc. We encourage students and composers to conceive of their own ways to interface the players with the computers, and we have provided for the ability to connect a wide variety of devices including custom ones quickly and easily. We also have a variety of microphones (handheld and headset) and pickups that can be used bringing in live sound to each instrument.

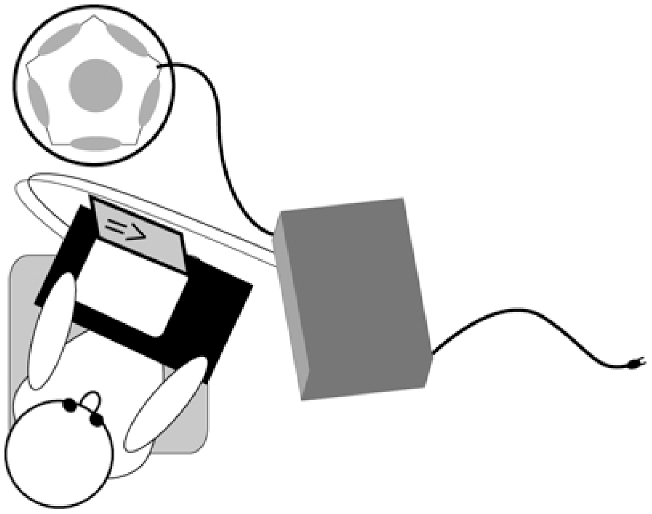

Figure 2.1. A PLOrk Meta-Instrument

Figure 2.2. PLOrk rack.

The six-channel speaker, as mentioned above, consists of six individually addressable drivers; each driver can be accessed via a direct path from the computer. The interconnection of six channels of amplification to this speaker is accomplished with a special cable consisting of 6 1/4" TRS plugs on one end, going to a Souriau Trim-Trio circular multipin connector. In addition to the six-channel speaker, we have four subwoofers (the Sunfire True Subwoofer Super Junior) that can be interfaced with select instruments to provide for representation of low frequencies.

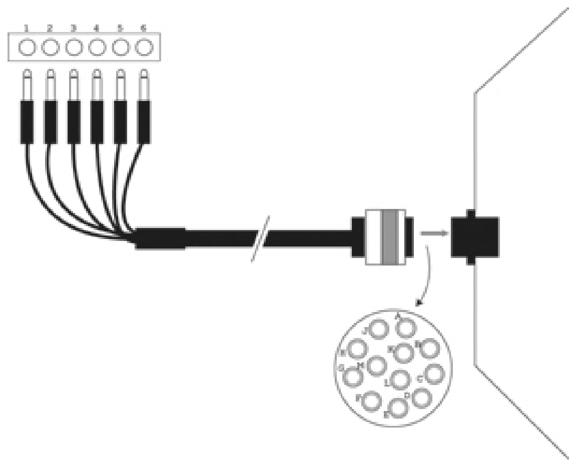

Fig 2.3. Speaker Cable Assembly

Each player sits on a meditation pillow and either holds the laptop literally on his/her lap (supported and protected by a lap-desk), or places the laptop on the rack to the right and holds instead some interface to the laptop, depending on the requirements of the composition. The speaker sits directly in front of each performer. In this way, each instrument is completely self-contained.

2.2 The Ensemble

The ensemble consists of 15 such meta-instruments organized into a specific pattern (for most performance spaces), derived after some initial experimentation. We found this configuration to be maximally efficient in terms of space, ability to hear oneself in relation to others, and for assigning roles to the ensemble. Each "seat" has a position indicator, which identifies the performer and his/her role within the composition. This is the default configuration of PLOrk, with four across the front (A,B,C,D), two in the center (X,Y), and nine in an arch surrounding the others (1-9).

PLOrk players are able to communicate with each other or with a conductor through networking protocols (Wright, Freed, Momeni 2003) which can be programmed within the performance environment in Max/MSP or ChucK (see below). We have been able to successfully control workstations, send text messages, scores and other notations, and lock the entire ensemble with tight tempo controls, all using the wireless network. Several pieces have taken advantage of networking structures (see descriptions of individual pieces from the first concert below). We have also experimented with audio networking by linking workstations together using spare audio ins and outs on the audio interface.

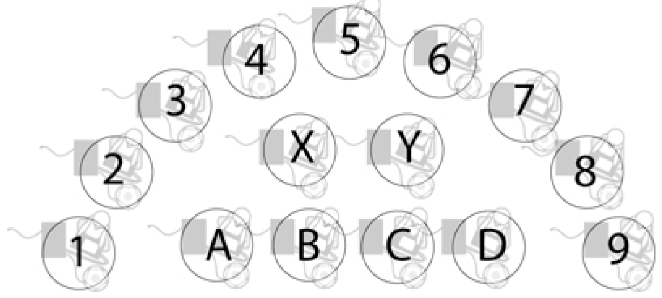

Fig 2.4. PLOrk Ensemble Layout

2.3 Software Tools

For much of the software development in PLOrk we have used the Max/MSP programming environment as well as ChucK (Wang and Cook 2003), a programming language for strongly-timed, on-the-fly sound synthesis and computer music performance currently in development at Princeton. The languages are used in complementary manner both for teaching and for performance. The ensemble has performed pieces created and run in one or even both of these environments, as well as other software environments, including SuperCollider and custom software applications in Java written specifically for PLOrk.

In order to facilitate some of the common programming needs in PLOrk, we have built a library of abstractions (the PLOrk Utilities) in Max/MSP that can be copied and tailored towards the needs of a particular instrument or piece. The PLOrk Utilities include:

1. a set of mapping abstractions that allow the composer to easily map a control signal to a signal processing or synthesis parameter. These control signals might be simple one-to-one controls (i.e. pressure) or more abstract, "second-order" controls (i.e. graphics tablet X-position * graphics tablet Y-position * pen pressure). These abstractions allow the user to specify parameter ranges, non-linear warping of the control signal, and smoothing, and can operate at the control rate or signal rate. All the mapping settings can be stored using the new Max/MSP "pattrstorage" system.
2. a front-end for the "pattrstorage" system—called "pattrface"—that allows users to quickly save, name, recall, and sort presets for compositions.

3. network utilities that enable a variety of different kinds of communications. Some simply allow a "conductor" computer to send messages to all the machines, while others can query the machines to determine which patches are running, and then divide the messaging accordingly. One utility allows the conductor machine to determine which machine is sitting at which location, for location-specific networking.

Our Max/MSP installation also includes a large number of widely used third-party externals.

ChucK is a real-time audio programming language that provides precise control over time and a dead-simple concurrent programming model. The language is strongly-typed, text-based, and provides a syntax that is both clear to write and to read. ChucK supports MIDI, OSC, as well as on-the-fly programming (Wang and Cook 2004). The expressive power of the language and its ease of use make it a useful tool for building software and pieces for PLOrk, and its gentle learning curve makes ChucK suitable for teaching.

Software developed as part of PLOrk includes low-latency network utilities over OSC that provide communication between ChucK/Max to any other OSC-enabled hosts, frameworks for rapid-prototyping, and a suite of programs for teaching the language, programming, and sound synthesis.

2.4 Communication and Networking

Often, the ensemble requires low-latency synchronization among all or some subset of the machines. As mentioned earlier, we have developed software frameworks in both Max/MSP and ChucK to synchronize PLOrk hosts over OpenSound Control. These frameworks can be easily integrated into pieces requiring synchronization, text/score message passing, and general data transfer of any type. The networking is supported over a 802.11g, 54 Mbps wireless LAN (using an Apple Airport Extreme Base Station). Using this setup, we can synchronize all 15 hosts with 30-40ms latency or better (measured approximately). Given that the orchestra is a bit over 40 feet wide, this latency situation is really the same or less than the acoustic latency that exists from one side of the orchestra to the other due to the speed of sound. From a musical performance perspective, the latency feels more-or-less normal, given the familiar separation within larger ensembles, and we claim that PLOrk can synchronize a pulse at a faster tempo with a network than a comparably sized non-networked percussion group could (it would, however, be thrilling to be proved wrong!).

2.5 Spatialization

Because the speaker is a six-channel instrument, rather than a front-firing speaker cabinet, it provides for interesting possibilities in terms of modeling sound radiation patterns of instruments real or imagined. Essentially we have access to 90 (15x6) individual point sources and 4 subwoofer channels, an unusual spatialization paradigm, with an emphasis not on a "surround" experience, but on a large field of sonic display with the same kind of spatial dimension as an orchestra of acoustic musicians whose instruments send sounds out from their resonating bodies in 360 degrees. For example, one can create a group texture in which each individual instrument is contributing to one component of a larger texture, with independent control over that component. Or, each individual instrument can be treated as a discrete voice as in a traditional acoustic ensemble. In any case, this model has significant social consequences, allowing one to spatially associate a sound source with a specific person.

In addition to the live performance model in the concert hall, there are also interesting possibilities for non-traditional performance spaces and/or installation pieces. For example, in May of 2006 we presented a concert of pieces in the Chancellor Green Rotunda at Princeton, which features a circular balcony that encircles the dome-shaped room above. We arranged the plorkestra on this balcony, around and above the audience. The spatial dimensions that this arrangement provided were unique and fun to work with.

.

3 PLOrk Compositions

Composing for PLOrk is an enormous challenge. Not only do we have the familiar challenges of composing for a large ensemble (and PLOrk does indeed *feel* "orchestral" in scope), we also have to invent the instruments for the players to play, teach them how to play them, and work out how the composition will be coordinated. Our experience has been so far that every composer who comes to work with PLOrk for the first time goes away deciding to start over again; the first PLOrk encounter can be completely shocking.

Determining how the notion of the "conductor" is conceived is central to any PLOrk composition. Do we have a conductor in the traditional sense, using familiar physical conducting patterns (sometimes this is the best solution), or do we work with the network? If we work with the network, at what level? Do we use the network to provide coarse communications (text messages, for instance), or to actually control the timing of the ensemble, allowing the players to work at a higher, non-event level with their instruments? Or do we dispense with the notion of a conductor entirely? All are possible, sometimes simultaneously.

Following are descriptions of pieces that PLOrk performed in its very first concert in January 2006; recordings of all these performances are available on the PLOrk website (plork.cs.princeton.edu). In addition to these works, new pieces, by Paul Lansky, Brad Garton, Dan

Trueman, Curtis Bahn and Tomie Hahn, and others, were performed in April of 2006, featuring guest performances by the tabla virtuoso Zakir Hussain, accordionist Pauline Oliveros, and the percussion quartet So Percussion; these works will be discussed in a future publication.

3.1 *The ABC Song*

With one simple instrument, designed by Trueman, we explore making music with the built-in QWERTY keyboard on the laptops by having each student record themselves speaking the name of each key multiple times. Using a simple program to build the instrument, each player presses each key a number of times while simultaneously speaking the name of that key; in a single session, the program records the utterances into a single long soundfile while also building a lookup table of start times associated with each key. The players can then go in and "tune" the instrument, assuring that the recorded times are accurate and responsive, without cutting off the beginnings of each sample. For the alphabet, with about 5 samples per key, such a sampling session typically takes about 20 minutes, including the "tuning."

Now sonified, the keyboard becomes a musical instrument it its own right, one that builds on established typing technique and is personal to each player. When a particular key is struck, the instrument randomly chooses one of the multiple samples recorded for each key (typically around 5 samples per key) and plays it back directly, or through a variety of filters. The duration of each sample is controlled by how long the key is held. For the *ABC Song,* the samples are fed through a set of tuned comb filters, arranged in such a way so that when the alphabet is typed, the pitches of the familiar ABC song are imparted on the spoken letters via the comb filters. The *ABC Song* is performed with a traditional conductor in three-part harmony, with PLOrk divided into sections.

3.2 *The PLOrk Drones*

This piece, by Dan Trueman, is a quasi-improvisation inspired by the so-called "Risset-Arpeggio." In the original Risset Arpeggio (Risset 1985), a set of oscillators with low fundamentals (around 60Hz) and primarily high partials are detuned ever so slightly to create beating patterns between the overtones. When synthesized by a single computer, a slowly ascending and descending arpeggio of the overtones emerges. In "The PLOrk Drones," each player is given a single such oscillator and a mechanism for controlling its overtones and fundamental frequency (within a very small range). Some players use accelerometers, others use graphics tablets, and others just simple sliders as controllers. A conductor machine sends texts messages to the various players to steer the piece, and can also impart large pitch changes to the drones. The plorkestra improvises elements within a group texture, at times attempting to maximize or minimize the beating patterns. While the familiar arpeggio vanishes—there is simply too much going on with phase and the multiple speakers of PLOrk to allow the arpeggio to emerge—the beating patterns are nonetheless ever present and "performable."

3.3 *On The Floor*

"On the Floor," by Scott Smallwood, is a piece whose sound is a side-effect of the process of turning the ensemble into a group of individual gamers. The first in a series of pieces to explore gaming and individual representations of similar sounds, this piece recreates the soundscape of an Atlantic City casino. Written entirely in ChucK, each instrument is a virtual slot machine. Each player begins with a certain number of credits, and simply plays the game until he or she is out of money. The program emulates the sound of a slot machine, but after a threshold is reached, the sound world changes, becoming more and more abstract. So, as players begin to lose money, the soundscape changes from being a specific place to being a sonic abstraction of that space. Strategies exist for staying in the game longer by betting more or less credits. If more credits are bet each round, the odds are slightly less, but the payoff can be much more. The conductor has the ability to surveil the group, and to affect the odds of any specific player. In this way, the conductor has the ability to extend or shorten the length of the piece by keeping tabs on players who are winning or losing too much.

3.4 *Non-specific Gamelan Taiko-Fusion Band*

This piece, written in ChucK/Audicle by Perry Cook and Ge Wang, is an experiment in human controlled, but machine synchronized percussion ensemble performance. A single "conductor" computer controls timing, while various percussive sounds are temporally positioned by PLOrk members. The piece gradually transitions from tuned bell timbres to drums as the texture and density grows. Orchestra members also have physical percussion instruments, and at times play those in human synchrony to the machine percussion parts. The "score," consisting of color-coded instrument and density guidelines, emerges from a computer printer in real time. The human conductor holds up these sheets as they emerge, and the plorksters busily go about modifying parameters and performance. In some performances, PLOrk is joined by traditional hand-drummers, creating a kind of network-mediated drum circle.

3.5 *The PLOrk Tree*

This piece, by Dan Trueman, is a quasi-improvisation based on a network binary tree. The parent node of the tree (position 5) acts as a kind of conductor, feeding the network various information which percolates through the tree and is ultimately fed back to the parent node.

Locked to a common pulse, the PLOrk members control a group texture by inheriting information from a network neighbor, and then making slight (or not so slight) modifications to that information, including pitches, timbres, and text messages, which are then all sent on to two other network neighbors, eventually feeding back through the tree; all of the information from G3 (the group furthest into the tree; see Fig. 4.5) is sent back to node 5, and the conductor at node 5 can pick and choose from that data and feed it back again into the network.

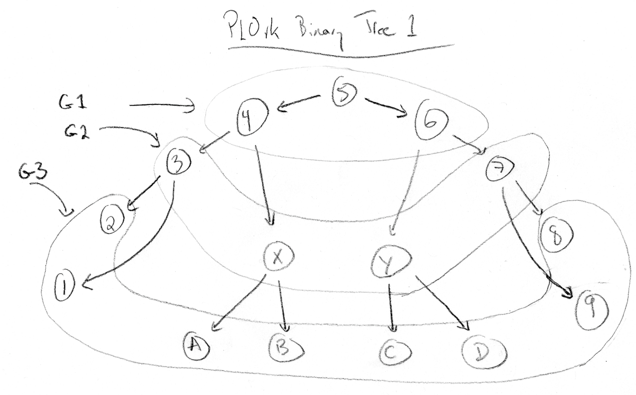

Fig 4.5: the binary tree used in the PLOrk Tree. Circled groups G1, G2, G3 represent the three "depths" of the tree network and are addressable as groups by the conductor.

4 PLOrk Problems

Some of the challenges we were presented with during our first year were predictable, others unexpected. They ranged from technical challenges to aesthetic ones, as well as the human personnel challenges that always exist in musical ensembles.

One of the biggest technical challenges involved random and unexplained network packet drops. While a large percentage of our networking problems turned out to be coding or user errors, we have attributed many of our problems to the instabilities of wireless networks, which we found were susceptible to interferences of various kinds, particularly when other networks were present. One way of combating this problem was to build in a certain amount of redundancy into our software systems, particularly for critical timing messages. Another method was to avoid the trap of overusing the network, and to actually rely on traditional conductor roles for critical communications, like hand signals, gestures, and even holding up cards. For the most part, though, our use of networking protocols like OSC were successful in most environments. In the future we hope to look into base-stations with stronger signals, and explore strategies for making our the base stations impervious to other local wireless traffic. We also plan to have a wired router on hand for situations that prove intractable (for instance, we encountered serious timing issues in a performance at Dartmouth College that we were unable to resolve wirelessly). We should also point out that most of the time the wireless network functioned well and allowed precise timing control with minimal packet drops.

Another issue that continues to challenge us is that of file synchronization. We maintain the machines as exact clones of each other, and this presents problems, particularly during the development of software for new pieces. It is necessary that each machine be the same for a variety of reasons. One reason is a practical one: since machines are often checked out by students or faculty during the week, it is important that their files get propagated to all of the machines before each rehearsal. Often we are developing and debugging files that must be available to the entire ensemble, and therefore we maintain a tight schedule of file syncing each week. Another reason has to do with the reality of machine failure. If a machine goes down, we need to be able to quickly replace that machine with another. We are still searching for the right utility and methodology to make this as seamless and flexible as possible. We currently use CVS for synchronization, which works well most of the time, but sometimes causes problems with filetypes in the OSX filesystem, and also has difficulty with large synchronizations (when soundfiles are involved, for instance).

We have invested significantly in portability for the ensemble, because it is important for us to be able to take the equipment out of the music building and into the concert hall, both on and off campus. Many years of hauling personal gear to professional gigs in often inhospitable environments has informed our decisions, but even so we experienced some tragedies with damaged equipment, particularly computers. Several machines were damaged in the same location (the corner near the power adapter input), and this prompted us to rethink the way the machines themselves were ported to and from rehearsals and venues. We ended up investing in a large "laptop vault" which houses all of our laptops in a single padded case on wheels, and this has turned out to be one of our best investments.

There are many opportunities for things to get damaged, especially since we do not have a permanent, dedicated rehearsal space. Our rehearsal space is shared with other ensembles, and it is not possible for us to keep the group set up for more than three hours every week. So, we must cart all of the equipment up and down two floors of the music building, from our storage room to the rehearsal hall, every week. Our facilities include a lab where we keep four workstations up and running so that we can test networking and other software issues with more than one machine, and where we can have sectional rehearsals. But in the future we hope to have a dedicated facility that would enable us to leave the orchestra set up, thus making it possible to have more development time and more rehearsals.

We have also had our share of problems that are typical of any large ensemble: getting and holding the attention of the players during rehearsals! Obviously, it's fun to make noises with computers, and just like acoustic instruments, the impulse to "wank around" is strong. We have often resorted to a networking means of silencing the group by turning off the DSP engine remotely on all of the machines, but more commonly we have tried to teach the players to observe the "plork silence" command or hand signal. In addition, there is the problem of technical difficulties taking a player out, either during software tutorials or rehearsals. One way we have chosen to deal with this is to teach the players how to troubleshoot their own instruments. This is really the first step in training players to be effective members of the ensemble; at the very beginning we teach them how things are connected physically, how the sound interfacing works, how to break down a problem and solve it own their own whenever possible. But still, sometimes there are issues that simply cannot be overcome easily, and so rehearsals usually involve a conductor/director, plus one or two people who run around and assist individuals with problems. Again, hand signals are often used to communicate that assistance is needed (thumbs down) or that everything is working fine (thumbs up). For those who have performed regularly with laptops, troubleshooting is a familiar skill, but these issues become greatly amplified when dealing with 15 systems and players who are inexperienced. In the future, we hope to establish an ongoing community of players with common knowledge of how to keep things running,

These problems require particular attention when preparing a performance. In addition to simply rehearsing the pieces, a significant amount of time is required to rehearse and document the transitions from piece to piece; how to change the software, make adjustments to the hardware, setup whatever networking is needed, and so on. This places a substantial burden on the composers, requiring that they make their patches and programs streamlined so they can be loaded and initialized as quickly as possible.

5 PLOrk Lessons and Directions

PLOrk presents a variety of new compositional, performative, and technical challenges. From both electronic and acoustic composition vantage points, the ensemble looks both familiar and radically new. Acoustic composers accustomed to working with larger ensembles might find the challenge of composing for 15 musicians undaunting, but might be overwhelmed by the variables and uncertainties presented by the PLOrk meta-instruments (there is no orchestration/instrumentation book for PLOrk!). Electronic music composers experienced working in a studio might likewise by uncertain with how to create interactive instruments. Those who are familiar with such designs may have to rethink their approaches to scale them to such a large ensemble (an instrument that works well solo, or in duos and trios, might not fare well at all in a larger ensemble), or have minimal experience working with so many musicians. Even those composers skilled in both electronic and acoustic composition find themselves in new territory, never having had to simultaneously negotiate challenges that typically remain in separate domains.

PLOrk is in a sense an experiment in parallel processing, both silicon-based (the laptops) and carbon-based (the musicians). It is certainly possible to automate virtually all aspects of a piece and have the players simply turn their instruments on and leave. Likewise, it is obviously possible to simply ignore the laptops and have the musicians play acoustically (some of our pieces do have the performers working with purely acoustic instruments simultaneously with the PLOrkstations). Determining where between those extremes we are at any particular point in a piece is a constant question; sometimes it is useful to have the players making simple adjustments (turning on/off processes, controlling levels) while other aspects are automated or controlled via the network, while at other times it is preferable to make full use of each player's bandwidth. And, as with all things PLOrk, this determination may have to be made 15 times (or more), depending on how the ensemble is sectionalized. In a future publication we will explore these ideas more deeply through some of the compositions currently in progress.

Our experiences with PLOrk in this first year have been encouraging. The challenges are compelling and we are convinced that there is much interesting music to be found and made with PLOrk, music never before possible or even imaginable. Pending funding support, we are hopeful that PLOrk will become a regular ensemble at Princeton, along with the conventional orchestra, jazz ensembles and choirs. We envision creating a curriculum around the ensemble that prepares students for both the technical and musical challenges posed by PLOrk. Such a curriculum would also provide a general introduction to computer music and audio technology.

The future of PLOrk will be largely determined by those composers who compose for it and we hope to involve many composers over the coming years. In addition, we hope to take PLOrk on the road (for our first trip, we performed at Dartmouth College in May 2006) to perform in different spaces and for different audiences. Finally, we plan to compose more chamber music, using just a few of the PLOrk meta-instruments at a time; chamber music contexts with laptops look quite different after dealing with a large ensemble like PLOrk.

`http://plork.cs.princeton.edu/`

6 Acknowledgments

We would like to acknowledge the many sources of funding that enabled PLOrk to take flight, including the following Princeton University organizations: the Council on Science and Technology, the Freshman Seminar series, the Redistribution Initiative, the School of Engineering and Applied Science, the Music and Computer Science Departments. Also, thanks to Apple Computer. Finally, thanks to the students of FRS 175, our first course centered around PLOrk, for their patience and energy.

7 References

Bahn, C. and D. Trueman, “interface—electronic chamber enesmble,” CHI 2001 Workshop on New Interfaces for Musical Expression, `http://www.informatik.uni-trier.de/~ley/db/conf/nime/nime2001.html`.

Bahn, C., T. Hahn and D. Trueman, “Physicality and Feedback: A Focus on the Body in the Performance of Electronic Music,” Proceedings of the International Computer Music Conference, Havana Cuba, 2001.

Cook, P. and G. Scavone, "The Synthesis ToolKit (STK)," International Computer Music Conference, Beijing, October, 1999.

Cook, P. and D. Trueman, "NBody: Interactive Multidirectional Musical Instrument Body Radiation Simulations, and a Database of Measured Impulse Responses," Proceedings of the International Computer Music Conference, Ann Arbor 1998.

The Hub, http://en.wikipedia.org/wiki/The_Hub_(band).

Mimio, http://www.gjp.info/infomimeo.htm.

McCartney, J. ”SuperCollider: A New Real-time Synthesis Language,” Proceedings of the International Computer Music Conference, Hong Kong, August 1996.

Puckette, M. “Combining Event and Signal Processing in the MAX Graphical Programming Environment,” Computer Music Journal, Vol 15, No. 3, 1991.

Risset, J.C., ``Computer music experiments, 1964--,'' *Computer Music Journal*, vol. 9, Spring 1985.

Trueman, D., C. Bahn, P. Cook, “Alternative Voices for Electronic Sound,” Proceedings of the International Computer Music Conference, Berlin, October 2000.

Trueman, D. and P. Cook, "BoSSA: The Deconstructed Violin Reconstructed," Journal of New Music Research, 29:2, Fall, 2000.

Trueman, D. and R. L. DuBois, “PeRColate” http://www.music.columbia.edu/PeRColate/ 1997–

Wang, G. and P. Cook, "ChucK: A Concurrent, On-the-fly, Audio Programming Language," Proceedings of the International Computer Conference, Singapore, Oct. 2003.

Wang. G. and P. Cook. “On-the-fly Programming: Using Code as an Expressive Musical Instrument,” Proceedings of the International Conference on New Interfaces for Musical Expression, Hamamatsu, Japan, June 2004.

Wessel, D. "Instruments That Learn, Refined Controllers, and Source Model Loudspeakers," Computer Music Journal, Vol. 15, No. 4, Winter 1991.

Wright, M., A. Freed, A. Momeni, “OpenSound Control: State of the Art 2003” New Interfaces for Musical Expression, Montreal, Canada 2003.