A conditional model for automatic orchestration Application to a real-time Live Orchestral Piano

LEOPOLD CRESTEL, Institut de Recherche et Coordination Acoustique/Musique
PHILIPPE ESLING, Institut de Recherche et Coordination Acoustique/Musique

Multifrequency media access control has been well understood in general wireless ad hoc networks, while in wireless sensor networks, researchers still focus on single frequency solutions. In wireless sensor networks, each device is typically equipped with a single radio transceiver and applications adopt much smaller packet sizes compared to those in general wireless ad hoc networks. Hence, the multifrequency MAC protocols proposed for general wireless ad hoc networks are not suitable for wireless sensor network applications, which we further demonstrate through our simulation experiments. In this article, we propose MMSN, which takes advantage of multifrequency availability while, at the same time, takes into consideration the restrictions of wireless sensor networks. Through extensive experiments, MMSN exhibits the prominent ability to utilize parallel transmissions among neighboring nodes. When multiple physical frequencies are available, it also achieves increased energy efficiency, demonstrating the ability to work against radio interference and the tolerance to a wide range of measured time synchronization errors.

Additional Key Words and Phrases: Automatic orchestration, Real-time, Conditional Restricted Boltzmann Machine, time modeling

ACM Reference Format:
Léopold Crestel, Philippe Esling, 2010. A conditional model for automatic orchestration
Application to a real-time Live Orchestral Piano. *ACM Trans. Embedd. Comput. Syst.* V, N, Article A (January YYYY), 12 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Musical orchestration is the subtle art of writing musical pieces for orchestra, by combining the spectral properties specific to each instrument in order to achieve a particular sonic goal. This complex discipline involves a wide set of intricate mechanisms, most of which have not yet been satisfactorily theorized. Indeed, famous composers often conjectured that orchestration would mainly remain an empirical discipline, which could only be learned through experience and never axiomatized in books. Even if several famous musicians have written orchestration treatises [Berlioz 1844; Koechlin 1941], those mostly remain recommendations and sets of existing orchestration examples from which one can draw inspiration. We focus more specifically in this work on *projective* orchestration, which is the transformation from a piano score to an orchestral piece figure 1 on page 2. Many composers have worked in a projective manner, and a large amount of examples can be found in the repertoire. For instance, one of the most famous is the orchestration of *Les tableaux d'une exposition*, a Modest Moussorgsky's piano piece, by Maurice Ravel.

This work is supported by the National Science Foundation, under grant CNS-0435060, grant CCR-0325197 and grant EN-CS-0329609.
Author's addresses: L. Crestel and P.Esling, Représentation Musicales, Institut de Recherche et Coordination Acoustique/Musique;

DOI: http://dx.doi.org/10.1145/0000000.0000000

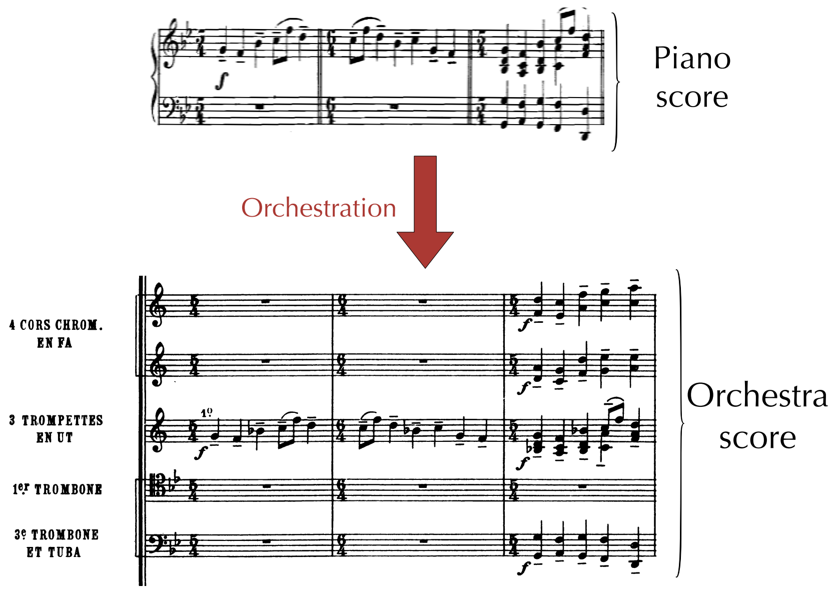

Fig. 1. *Projective orchestration.* A piano score is extended (projected) on an orchestra. For one piano score, many acceptable orchestration exist. Our hypothesis is that a piano score is strongly correlated to any of the orchestration that could be produced from this piece.

The objective in this work is to be able to automatically perform in real-time the *projective* orchestration of a piano performance. More specifically, our system takes in input a piano score and outputs an orchestral score. The vast combinatorial set of instrument possibilities added to the complex temporal structure of polyphonic music make this problem a particularly daunting task. Several attempts to build an automatic orchestration system can be found in the literature.

Orchestration can be viewed as assigning the different notes of the piano score to a certain number of instrument according to constraints over the number of instruments, their tessitura or a certain voice leading. Interpreting orchestration as a Constraint Solving Problem (CSP) lead to a first solution [Truchet and Assayag 2011]. However, as Steven McAdams pointed it out, timbre is "a structuring force in music" [McAdams 2013] in the sense that it should be used to emphasize the already existing structure of the original piano piece. We believe that a system built only on symbolic constraint will undoubtedly fail at grasping the harmonic, rhythmic and melodic structure of the original piano piece and thus propose an interesting orchestration. *Orchids* ([Esling et al. 2010]) is an other interesting work set in an other paradigm called *injective* orchestration. It consists in trying to reconstruct a target timbre for a small temporal frame. Its major drawback is that *Orchids* can only orchestrate short frames (less than 10 seconds). In order to build an automatic orchestration system being able to work on a macro-temporal timescale while structuring the musical discourse, statistical inference appeared us to be a promising solution. Their should indeed exist strong correlation between the information contained in the original piano score and the orchestral rendering we want to produce. Statistical inference would allow us to extract rules from the vast knowledge embodied in the many orchestration proposed by famous composers over the years.

We decided to work with a class of models called conditional models [Taylor 2009], which derive from a particular type of Markov Random Field called the Restricted

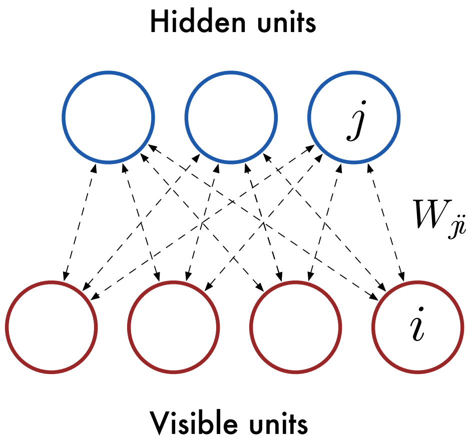

Fig. 2. *Restricted Boltzmann Machine*. The RBM is an energy-based model. Its energy function is computed from the values of weights that link nodes. Training an RBM consists in lowering the energy function around the example from a training set. Inference in this model is easy to perform since the hidden (resp. visible) units are independent from each others.

Boltzmann Machine (RBM) [Fischer and Igel 2014]. While being able to model complex distributions through latent units, those models implement a notion of context which allow us to model the influence of the past over the present and of the piano over the orchestra. Those models are generative which is a requirement in our case. If correctly trained on a training dataset, a model then has the ability to generate data that are similar, yet unseen, to those contained in the training set. This is through this mechanism that orchestral projection can be performed.

We then propose in this article a new evaluation framework for the orchestral projection task in order to evaluate the different model previously introduced. This evaluation rely on a frame-level predictive task based on an accuracy measure. The results of the proposed model are then presented. We picked out the best model and included it in a real-time orchestration system called *LOP*.

This paper is organized as follows. In sections 2 we introduce the state of the art in conditional models through three well known models: the RBM, the CRBM and the FGCRBM. The orchestration projection task is presented in the section 3 along with an evaluation framework based on a frame-level accuracy measure. The previously introduced models are then evaluated in this framework and the results displayed. The section 4 introduces a real-time *projective* orchestration system using the presented architectures.

2. STATE OF THE ART

The three model we have used in our work are the Restricted Boltzmann Machine (RBM), the Conditional RBM (CRBM) and the Factored Gated Conditional RBM (FGCRBM). They derive from the Graphical Probabilistic Model (GPM) theory and more specifically from a class of models called Markov Random Fields (MRF) ([Fischer and Igel 2014]). They are presented by increasing level of complexity, each model adding a new *degree of freedom* to the previous one.

2.1. Restricted-Boltzmann Machine

A Restricted-Boltzmann Machine (RBM) [Hinton et al. 2006] is an energy-based model that represent the joint distribution of a visible vector $\boldsymbol{v} = (v_1, ..., v_m)$ and a hidden vector $\boldsymbol{h} = (h_1, ..., h_n)$. This distribution is given by $p(\boldsymbol{v}, \boldsymbol{h}) = \frac{\exp^{-E(\boldsymbol{v},\boldsymbol{h})}}{Z}$ where

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_{i=1}^{m} a_i v_i - \sum_{j=1}^{n} b_j h_j - \sum_{i=1}^{m}\sum_{j=1}^{n} v_i W_{ij} h_j \tag{1}$$

and $Z = \sum_{v,h} \exp^{-E(v,h)}$ is a usually intractable partition function. $\boldsymbol{\Theta} = \{\boldsymbol{W}, \boldsymbol{a}, \boldsymbol{b}\}$ are the weights of the network. Unfortunately, the gradient of the negative log-likelihood of a vector from the training database $\boldsymbol{v}^{(l)}$ is intractable because of the negative term in right-hand part of the equation (2) which involves a sum over all the possible combinations of the hidden units (alternatively all the possible configurations of the visible units). A training algorithm called Contrastive divergence (CD) [Hinton 2002] rely on an approximation of the model driven term of this equation by running a k-step Gibbs chain to obtain a sample $\boldsymbol{v}^{(l,k)}$ (2).

$$-\frac{\partial \ln(p(\boldsymbol{v}^{(l)}|\boldsymbol{\Theta}))}{\partial \boldsymbol{\Theta}} = \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{v}^{(l)})}\left[\frac{\partial E(\boldsymbol{v}^{(l)}, \boldsymbol{h})}{\partial \boldsymbol{\Theta}}\right] - \mathbb{E}_{p(\boldsymbol{h},\boldsymbol{v})}\left[\frac{\partial E(\boldsymbol{v}, \boldsymbol{h})}{\partial \boldsymbol{\Theta}}\right] \tag{2}$$

$$\approx \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{v}^{(l)})}\left[\frac{\partial E(\boldsymbol{v}^{(l)}, \boldsymbol{h})}{\partial \boldsymbol{\Theta}}\right] - \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{v}^{(l,k)})}\left[\frac{\partial E(\boldsymbol{v}^{(l,k)}, \boldsymbol{h})}{\partial \boldsymbol{\Theta}}\right] \tag{3}$$

Running a Gibbs sampling chain consists in alternatively sampling the hidden units knowing the visible units then the visible units knowing the inferred hidden units by using the marginal probabilities (4).

$$p(v_i = 1|\boldsymbol{h}) = sigm\left(a_i + \sum_j W_{ij} h_j\right) \tag{4}$$

$$p(h_j = 1|\boldsymbol{v}) = sigm\left(b_j + \sum_i W_{ij} v_i\right) \tag{5}$$

where $sigm$ is the sigmoid function. Note that sampling from the marginal distribution is easy since visible units (respectively hidden units) are independent from each others. Hence, knowing the hidden units, all the visible units can be sampled in one step. This allows fast implementation through matrix operations and is known as *block sampling*. It has been proved [Bengio 2009] that the samples we obtain after an infinite number of iteration will be drawn from the joint distribution of the visible and hidden units of our model. An other approximation consists in starting the Gibbs chain from the sample $\boldsymbol{v}^{(l)}$, which increases the convergence of the chain, and to limit the number of alternate sampling steps to a fixed number K. After evaluating the statistics (model driven terms), the parameters can be updated. The whole algorithm is called Contrastive Divergence-K (CD-K). In a RBM those update rules are given by

$$\Delta W_{ij} =< v_i h_j >_{data} - < v_i h_j >_{model} \tag{6}$$

$$\Delta a_i =< v_i >_{data} - < v_i >_{model} \tag{7}$$

$$\Delta b_j =< h_j >_{data} - < h_j >_{model} \tag{8}$$

2.2. Conditional RBM

The Conditional Restricted Boltzmann Machine (CRBM) model ([Taylor 2009]) is an extension of the RBM. A dynamic bias is added to the static bias of the visible ($\boldsymbol{a}$) and hidden $\boldsymbol{b}$ units. This dynamic bias linearly depends on a set of unit called context units ($\boldsymbol{x}$). To model time series, if we consider that the visible units $\boldsymbol{v}(t)$ represent the current time frame, those context units can be defined as the concatenation of the N

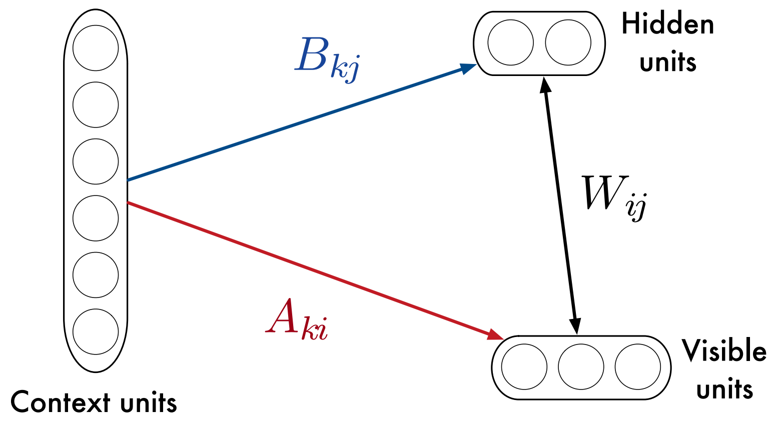

Fig. 3. *Conditional RBM*. A layer of context units is added to the standard RBM architectures. Those context units linearly modify the bias of both visible and hidden units.

last time frames $\boldsymbol{x}(t) = \left(v_1^{(t)}, ..., v_m^{(t)}, ..., v_1^{(t-N)} ..., v_m^{(t-N)}\right)$, where N denotes the order of the model. The energy function of the Conditional RBM is given by (9)

$$E(\boldsymbol{v}(t), \boldsymbol{h}(t)|\boldsymbol{x}(t)) = -\sum_i \hat{a}_i(t)v_i(t) - \sum_{ij} W_{ij}v_i(t)h_j(t) - \sum_j \hat{b}_j(t)h_j(t) \tag{9}$$

where the biases are defined by

$$\hat{a}_i(t) = a_i + \sum_k A_{ki}x_k(t)$$
$$\hat{b}_j(t) = b_j + \sum_k B_{kj}x_k(t)$$

This model can be trained by contrastive divergence, since the marginal probability of visible and hidden units are the same as in the RBM, while replacing the static biases by the dynamics biases. The following updates rules are obtained

$$\Delta W_{ij} =< v_i h_j >_{data} - < v_i h_j >_{model} \tag{10}$$
$$\Delta a_i =< v_i >_{data} - < v_i >_{model} \tag{11}$$
$$\Delta b_j =< h_j >_{data} - < h_j >_{model} \tag{12}$$
$$\Delta A_{ik} =< v_i x_k >_{data} - < v_i x_k >_{model} \tag{13}$$
$$\Delta B_{jk} =< h_j x_k >_{data} - < h_j x_k >_{model} \tag{14}$$
$$\tag{15}$$

2.3. Factored Gated Conditional RBM

The Factored Gated Conditional RBM model [Taylor and Hinton 2009] proposes to extend the Conditional RBM model by adding a layer of feature units $\boldsymbol{z}$ which modulate the weights of the conditional architecture in a multiplicative way. Hence, the weights of the networks become $\boldsymbol{\Theta} = \{\boldsymbol{W}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{a}, \boldsymbol{b}\}$, where $\boldsymbol{W} = (W)_{ijl}$, $\boldsymbol{A} = (A)_{ikl}$ and $\boldsymbol{B} = (B)_{jkl}$ are three dimensional tensors.

This multiplicative influence can be understood as a modification of the energy landscape of the model. Each configuration of the feature units defines a new energy function of the simple CRBM model defined by the other units ($\boldsymbol{v}$, $\boldsymbol{h}$, and $\boldsymbol{x}$). Since the

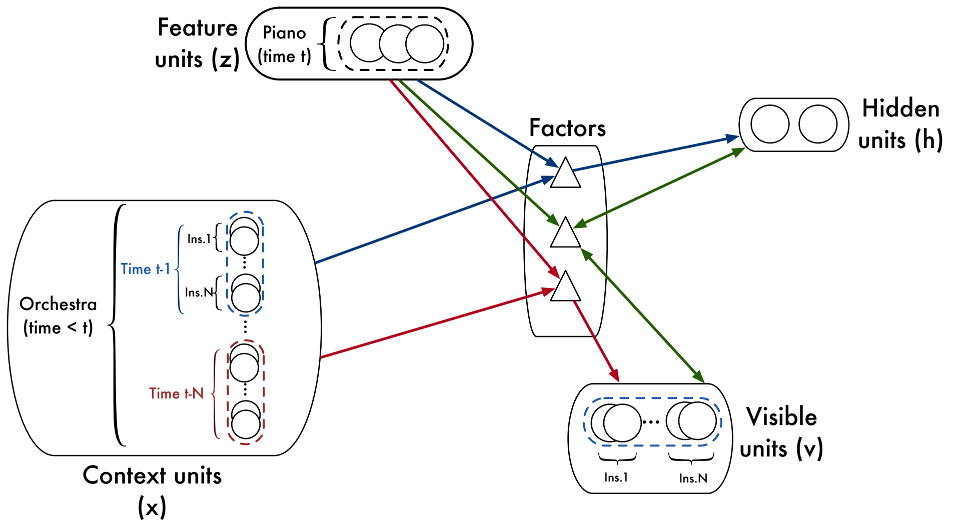

Fig. 4. FGCRBM model. The features units (bmz) modify the energy landscape of the model by a multiplicative influence over the weights $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{W}$. Here, the role of each units in order to perform orchestration is indicated.

number of parameters to train becomes high, the three dimensional tensors can be factorized into a product of three matrices by including factor units indexed by f : $W_{ijl} = W_{if}.W_{jf}.W_{lf}$. The energy function of this Factored Gated Conditional RBM is then given by

$$E(\boldsymbol{v}(t), \boldsymbol{h}(t)|\boldsymbol{x}(t), \boldsymbol{z}(t)) = -\sum_{f}\sum_{ijl} W^{v}_{if}W^{h}_{jf}W^{z}_{lf}v_i(t)h_j(t)z_l(t) - \sum_{i}\hat{a}_i(t)v_i(t) - \sum_{j}\hat{b}_j(t)h_j(t) \tag{16}$$

where the dynamic biases of the visible and hidden units are defined by

$$\hat{a}_i(t) = a_i + \sum_{m}\sum_{kl} A^{v}_{im}A^{x}_{km}A^{z}_{lm}x_k(t)z_l(t) \tag{17}$$

$$\hat{b}_j(t) = b_j + \sum_{n}\sum_{kl} B^{h}_{jn}B^{x}_{kn}B^{z}_{ln}x_k(t)z_l(t) \tag{18}$$

The FGCRBM model can be trained by contrastive divergence which lead to the following update rules for the parameter

$$\begin{aligned}
\Delta b^{(v)}_i &=< v_i >_{data} - < v_i >_{model}\\
\Delta b^{(h)}_j &=< h_j >_{data} - < h_j >_{model}\\
\Delta W^{v}_{if} &=< v_i \sum_j W_{jf}h_j \sum_l W_{lf}z_l >_{data} - < v_i W_{jf}h_j \sum_l W_{lf}z_l >_{model}\\
\Delta W^{h}_{jf} &=< h_j \sum_i W_{if}v_i \sum_l W_{lf}z_l >_{data} - < h_j \sum_i W_{if}v_i \sum_l W_{lf}z_l >_{model}\\
\Delta W^{z}_{lf} &=< z_l \sum_i W_{if}v_i \sum_j W_{jf}h_j >_{data} - < z_l \sum_i W_{if}v_i \sum_j W_{jf}h_j >_{model}
\end{aligned}$$

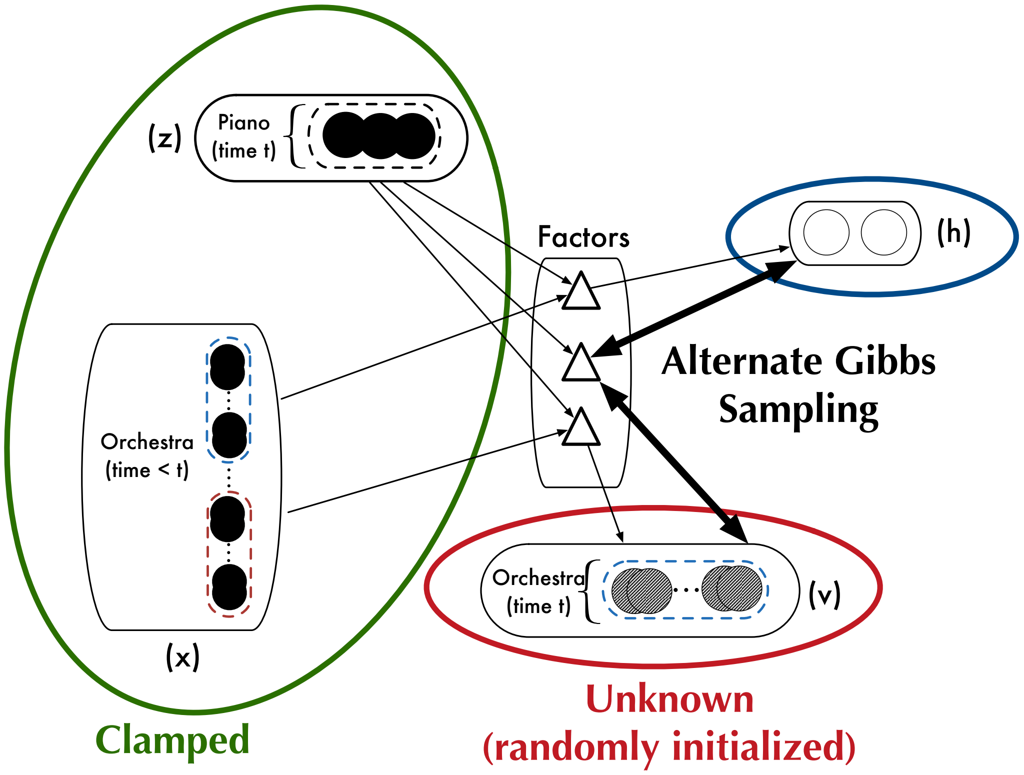

Fig. 5. *Sampling in a FGCRBM*. Context and Features units are respectively clamped to the last ($t-1$ to $t-N$) orchestral frames and the current (t) piano frame. Visible units are randomly initialized. Then, several Gibbs sampling step are performed, in our case 40.

$$
\begin{aligned}
\Delta A^{v}_{im} &=< v_i \sum_k A_{km} x_k \sum_l A_{lm} z_l >_{data} - < v_i \sum_k A_{km} x_k \sum_l A_{lm} z_l >_{model} \\
\Delta A^{x}_{km} &=< x_k \sum_i A_{im} v_i \sum_l A_{lm} z_l >_{data} - < x_k \sum_i A_{im} v_i \sum_l A_{lm} z_l >_{model} \\
\Delta A^{z}_{lm} &=< z_l \sum_i A_{im} v_i \sum_k A_{km} x_k >_{data} - < z_l \sum_i A_{im} v_i \sum_k A_{km} x_k >_{model} \\
\Delta B^{z}_{jn} &=< h_j \sum_k B_{kn} x_k \sum_l B_{ln} z_l >_{data} - < h_j \sum_k B_{kn} x_k \sum_l B_{ln} z_l >_{model} \\
\Delta B^{z}_{kn} &=< x_k \sum_j B_{jn} h_j \sum_l B_{ln} z_l >_{data} - < x_k \sum_j B_{jn} h_j \sum_l B_{ln} z_l >_{model} \\
\Delta B^{z}_{ln} &=< z_l \sum_j B_{jn} h_j \sum_k B_{kn} x_k >_{data} - < z_l \sum_j B_{jn} h_j \sum_k B_{kn} x_k >_{model}
\end{aligned}
$$

2.4. Generative models

The previously introduced models are generative. If a model has been correctly trained, its distribution approximates the underlying distribution of the training data. It is then possible to sample from this distribution in order to generate data that are similar to the data of the training set (figure 5 on page 7). The generation process can be described as follow. After randomly setting the visible units vector (for each index i, $\boldsymbol{v}_i \sim \mathcal{U}(0,1)$), alternate Gibbs sampling is performed in order to approximate the joint distribution of the hidden and visible units conditionally on the context units. Given the context units, one step of alternate Gibbs sampling consists in sampling the hidden units knowing the visible, then sampling the visible units knowing the hidden. In theory the visible sample obtained is from the model distribution after an infinite number of steps. If theoretically an infinite number of steps is necessary, in practice 20 to 100 steps are typically used. In our case, we obtained satisfying results with 40 sampling steps.

2.5. Approximations

In practice, several approximation are made both during training and generating phases. During the training phase, we have already seen the two approximations of the CD-K algorithm which consists in performing only a limited number of sampling steps, and starting the Gibbs chain with a sample from the training set.

2.5.1. Mean-field values. An other common approximation can be made both during training and generating phases, when running the Gibbs chain in order to estimate the model-driven terms. In order to diminish the sampling noise mean-field values can be used for the visible units. This is not possible for the hidden units since it would violate the information bottleneck which is the fact that hidden units can only transmit a limited amount of information to the visible units [Hinton 2010]. Indeed, hidden units can represent 2^n different states where n is the number of hidden units.

2.5.2. Number of sampling steps. It is noteworthy that during the learning phase, only a small number of sampling step are performed (1 in our case, rarely more than 10) in the contrastive divergence algorithm. This is because the objective is simply to modify the energy of our model in order to increase the training set's likelihood under the model distribution. Indeed, it has been shown that even a small number of sampling steps guarantee to increase a lower bound over this quantity [Bengio 2009]. When generating data, we really want to obtain a sample as close as possible to the distribution of the model and then a larger number of Gibbs sampling steps need to be performed.

3. PROJECTIVE ORCHESTRATION

Automatic orchestration suffers from the lack of quantitative evaluation framework. The different works on the domain mainly rely on qualitative evaluation (REF MAIS LESQUELLES ???). To our best knowledge, there has not been any attempt in the automatic orchestration field to define a task associated to a performance measure. We propose here a first attempt in order to fill this gap by defining the orchestration projection task. This task consists in projecting a piano sequence on an orchestra. More precisely, it consists at each time t to generate an orchestral vector knowing the piano frame $Piano(t)$ and the recent past of sequence of orchestral vectors $Orch(t-1), ...Orch(t-N)$. We define in this section an evaluation framework for projective orchestration based on a frame-level accuracy measure and then present the result of the previously introduced model in this framework.

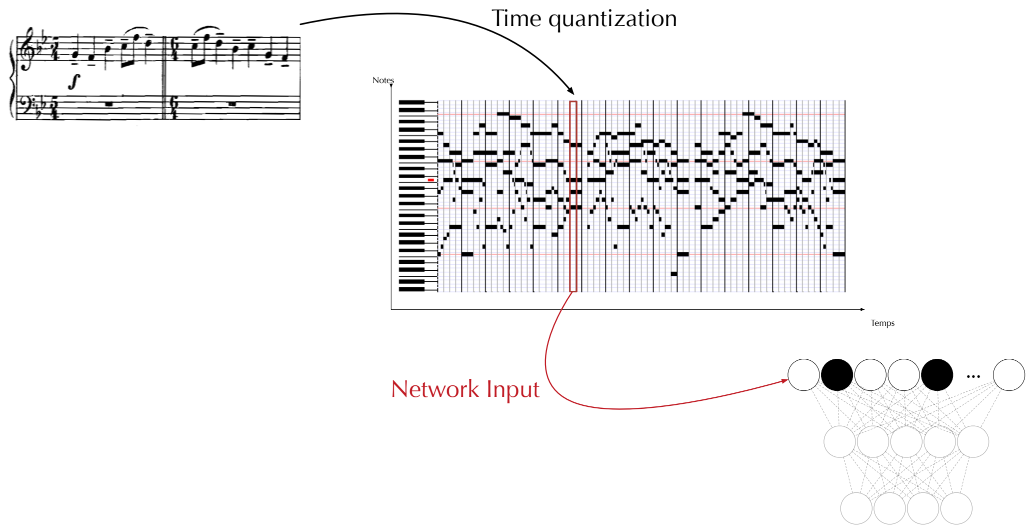

Fig. 6. *Data representation for a single piano*. The pianoroll is a representation of musical events, discrete on both frequency scale (pitch) and the time scale (frames). A pitch p at time t can be either on or off, which is represented by a one or a zero on the pianoroll. To represent an orchestra, the pianorolls of each instruments are simply concatenated along the pitch dimension.

3.1. Formalization

Conditional models allow to generate sequences of data under a certain context. Projective orchestration consists in producing an orchestral score conditionally on a piano score. In order to guarantee a certain continuity in the orchestration, voice leading for each instrument, the recent past of the orchestral sequence is also added in the contextual information.

3.1.1. Data representation. Before adapting the previously introduced models, the first step is to choose an adapted representation for the piano an orchestra scores. To model sequences of symbolic music, the pianoroll representation is often used. This is a binary matrix $Pianoroll(p,t)$ which specify for a certain time quantisation if a pitch p is played at time t. s The dynamics are ignored and each time frame is then represented by a binary vector $Pianoroll(t)$ This representation, usually defined for a single polyphonic instrument can easily be extended to an orchestra composed by N instruments by simply concatenating the pianoroll of each instrument over the pitch dimension.

$$Orch = \begin{bmatrix} Instrument\ 1 \\ Instrument\ 2 \\ \vdots \\ Instrument\ N \end{bmatrix} \tag{19}$$

Each instrument has a different playing range and the size of the pitch dimension is different for each instrument, limited to the pitch observed in the training dataset. For the piano score, the number of pitch is limited to 88 which is then number of keys on a piano keyboard. Note that we respect the usual simplifications used when writing orchestral scores which consists in grouping all the instruments of a same section. For instance, the section *violin 1*, composed by many instrumentalists (10 or more), is represented as a unique instrument. For a sequence of music of length T and with N instruments, $Orch$ is then a matrix of dimension $T \times \sum_{n=1}^{N} range(n)$.

In our framework we chose 14 instruments indexed by :

1. Violin	6. Timpani	11. Oboe
2. Viola	7. Trumpet	12. Bassoon
3. Cello	8. Trombone	13. Clarinet
4. Double-bass	9. Tuba	14.Flute
5. Harp	10. French horn	

3.1.2. Modeling orchestral sequences. This data representation can be directly used in the previously introduced conditional models. In the CRBM model, we consider that the visible units represent the orchestral vector for the time frame t, conditional units are be used to model the influence of the past orchestral vectors $Orch(t-1), ..., Orch(t-N)$ and the influence of the piano frame at time $Piano(t)$ over the visible units. The context units are then defined by the concatenation of the past orchestral frames and the current piano frame $Context(t) = \left[Piano(t)^T, Orch(t-1)^T, ..., Orch(t-N)^T\right]^T$.

The FGCRBM model allows to separate the influence of the current piano frame and the past orchestral frames. The current piano frame defines the feature units (z) $Features(t) = Piano(t)^T$, and the concatenation of the past orchestral frames define the context units (x) $Context(t) = \left[Orch(t-1)^T, ..., Orch(t-N)^T\right]^T$ (figure 4 on page 6).

One might be surprised that for each time frame t, we do not restrain the possible pitches of each instrument to the pitches seen in the piano score at the same time t. This i because orchestrating a piano score can not be reduced to the simple projection of the notes that are already written on the piano score. Instead, the harmonic structure is often densified by adding extra notes, from a simple octaviation to an harmonic expansion (more complex chords with a more specific colour), the melody might be doubled by several instruments at different octave or the rhythmic structure slightly modified (held note instead of a staccato).

3.2. Evaluation

Building a quantitative evaluation framework for generative models is rarely straightforward, especially since computing the likelihood of a test sample is intractable in the models we used. In music automatic generation, a common practice consists in defining an auxiliary task. The most frequently task is a predictive task based on a frame-level accuracy that can be easily extended to the orchestral inference task [Liu and Ramakrishnan 2014; Boulanger-Lewandowski et al. 2012; Lavrenko and Pickens 2003].

3.2.1. Frame-level accuracy. The frame level accuracy of a model over a testing set is defined as the mean value of the accuracy measured for each time frame. For each time frame, we try to predict the orchestral frame $\hat{Orch}(t)$ knowing the recent past $Orch(t-1), ..., Orch(t-N)$ and the piano frame $Piano(t)$ and compare it to the original frame $Orch(t)$. The accuracy measure the difference between the predicted and original frames [Boulanger-Lewandowski et al. 2012; Liu and Ramakrishnan 2014]

$$\text{Accuracy} = \frac{TP(t)}{TP(t) + FP(t) + FN(t)} \tag{20}$$

where $TP(t)$ is the number of notes correctly predicted (true positives). $FP(t)$ is the number of notes predicted which are not in the original sequence (false positive) and $FN(t)$ is the number on unreported notes (false negative).

Instead of binary values, activation probabilities are used for the predicted samples in order to reduce the sampling noise. In the case this probability is intractable, one should sample many predicted frames for each time frame and compute the mean value of those samples.

3.2.2. Event-level accuracy. A major flaw in the frame-level accuracy measure is its dependence to the rhythmic quantization chosen. Indeed, when the quantization become too small, it becomes highly probable that a frame is simply repeated at the next time frame. Hence, the best predictive model is simply a model which predict that the next time frame is the same as the frame $t-1$. This is not a desirable behaviour, and we propose an evaluation based on an event-level to address that issue.

A musical event is defined as a change in the orchestral score, either a note being switched on or off. The predictive event-level accuracy measure relies on the same accuracy measure previously defined (20). The only difference is that it occurs at each new event instead of each new time frame. Since the task is slightly different, the training phase has been consequently changed so that a model is trained only on new event frames.

3.3. Database

We used a parallel database of piano scores and their orchestration by famous composers. The database consists of 76 *XML* files. Given the complexity of the distribution we wanted to model and the reduced size of the database we have accessed to, we decided to keep as a test dataset only the last half of one track from our database. Hence 75 and a half files were used to train our model. We chose to do so in order to have the best generation ability. Mini-batch of size 100 have been used during the training phase. For the frame-level measure, 335 mini-batch were available, and 89 for the event-level measure. For each instrument, the pitch range is reduced to the tessitura observed in the training dataset. We used a rhythmic quantization of 8 frames per beat.

3.4. Results

4. LIVE ORCHESTRAL PIANO

5. CONCLUSION AND FUTURE WORKS

6. TYPICAL REFERENCES IN NEW ACM REFERENCE FORMAT

A paginated journal article [?], an enumerated journal article [?], a reference to an entire issue [?], a monograph (whole book) [?], a monograph/whole book in a series (see 2a in spec. document) [?], a divisible-book such as an anthology or compilation [?] followed by the same example, however we only output the series if the volume number is given [?] (so Editor00a's series should NOT be present since it has no vol. no.), a chapter in a divisible book [?], a chapter in a divisible book in a series [?], a multi-volume work as book [?], an article in a proceedings (of a conference, symposium, workshop for example) (paginated proceedings article) [?], a proceedings article with all possible elements [?], an example of an enumerated proceedings article [?], an informally published work [?], a doctoral dissertation [?], a master's thesis: [?], an online document / world wide web resource [?], [?], [?], a video game (Case 1) [?] and (Case 2) [?] and [?] and (Case 3) a patent [?], work accepted for publication [?], 'YYYYb'-test for prolific author [?] and [?]. Other cites might contain 'duplicate' DOI and URLs (some SIAM articles) [?]. Boris / Barbara Beeton: multi-volume works as books [?] and [?].

APPENDIX

In this appendix, we measure the channel switching time of Micaz [CROSSBOW] sensor devices. In our experiments, one mote alternatingly switches between Channels 11 and 12. Every time after the node switches to a channel, it sends out a packet immediately and then changes to a new channel as soon as the transmission is finished. We measure the number of packets the test mote can send in 10 seconds, denoted as N_1. In contrast, we also measure the same value of the test mote without switching channels, denoted as N_2. We calculate the channel-switching time s as

$$s = \frac{10}{N_1} - \frac{10}{N_2}.$$

By repeating the experiments 100 times, we get the average channel-switching time of Micaz motes: 24.3μs.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Maura Turolla of Telecom Italia for providing specifications about the application scenario.

REFERENCES

Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and trends in Machine Learning* 2, 1 (2009), 1–127.

Hector Berlioz. 1844. *Grand traité d'instrumentation et d'orchestration modernes*. Schonenberger.

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. 2012. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *arXiv preprint arXiv:1206.6392* (2012).

Philippe Esling, Grégoire Carpentier, and Carlos Agon. 2010. Dynamic Musical Orchestration Using Genetic Algorithms and a Spectro-Temporal Description of Musical Instruments. *Applications of Evolutionary Computation* (2010), 371–380.

Asja Fischer and Christian Igel. 2014. Training restricted Boltzmann machines: An introduction. *Pattern Recognition* 47, 1 (2014), 25–39.

Geoffrey Hinton. 2010. A practical guide to training restricted Boltzmann machines. *Momentum* 9, 1 (2010), 926.

Geoffrey E Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural computation* 14, 8 (2002), 1771–1800.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A Fast Learning Algorithm for Deep Belief Nets. *Neural Comput.* 18, 7 (July 2006), 1527–1554. DOI:http://dx.doi.org/10.1162/neco.2006.18.7.1527

Charles Koechlin. 1941. *Traité de l'orchestration*. Éditions Max Eschig.

Victor Lavrenko and Jeremy Pickens. 2003. Polyphonic music modeling with random fields. In *Proceedings of the eleventh ACM international conference on Multimedia*. ACM, 120–129.

I.-Ting Liu and Bhiksha Ramakrishnan. 2014. Bach in 2014: Music Composition with Recurrent Neural Network. *CoRR* abs/1412.3191 (2014). http://arxiv.org/abs/1412.3191

Stephen McAdams. 2013. Timbre as a structuring force in music. In *Proceedings of Meetings on Acoustics*, Vol. 19. Acoustical Society of America, 035050.

Graham William Taylor. 2009. *Composable, distributed-state models for high-dimensional time series*. Ph.D. Dissertation. University of Toronto.

Graham W Taylor and Geoffrey E Hinton. 2009. Factored conditional restricted Boltzmann machines for modeling motion style. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 1025–1032.

Charlotte Truchet and Gerard Assayag. 2011. *Constraint Programming in Music*. Wiley.

Received February 2007; revised March 2009; accepted June 2009

Online Appendix to:
A conditional model for automatic orchestration
Application to a real-time Live Orchestral Piano

LEOPOLD CRESTEL, Institut de Recherche et Coordination Acoustique/Musique
PHILIPPE ESLING, Institut de Recherche et Coordination Acoustique/Musique

A. THIS IS AN EXAMPLE OF APPENDIX SECTION HEAD

Channel-switching time is measured as the time length it takes for motes to successfully switch from one channel to another. This parameter impacts the maximum network throughput, because motes cannot receive or send any packet during this period of time, and it also affects the efficiency of toggle snooping in MMSN, where motes need to sense through channels rapidly.

By repeating experiments 100 times, we get the average channel-switching time of Micaz motes: 24.3 μs. We then conduct the same experiments with different Micaz motes, as well as experiments with the transmitter switching from Channel 11 to other channels. In both scenarios, the channel-switching time does not have obvious changes. (In our experiments, all values are in the range of 23.6 μs to 24.9 μs.)

B. APPENDIX SECTION HEAD

The primary consumer of energy in WSNs is idle listening. The key to reduce idle listening is executing low duty-cycle on nodes. Two primary approaches are considered in controlling duty-cycles in the MAC layer.

© YYYY ACM. 1539-9087/YYYY/01-ARTA $15.00
DOI: http://dx.doi.org/10.1145/0000000.0000000