AUTOMATIC CHANNEL ROUTING USING MUSICAL INSTRUMENT LINKED DATA

Nicholas Jillings and Ryan Stables

Digital Media Technology Lab
Birmingham City University
Birmingham, B4 7XG, UK
{nicholas.jillings, ryan.stables}@bcu.ac.uk

ABSTRACT

Audio production encompasses more than just mixing a series of input channels. Most sessions involve tagging tracks, applying audio effects, and configuring routing patterns to build sub-mixes. Grouping tracks together gives the engineer more control over a group of instruments, and allows the group to be processed simultaneously using audio effects. Knowing which tracks should be grouped together is not always clear as this involves subjective decisions from the engineer in response to a number of external cues, such as the instrument or the musical content. This study introduces a novel way to automatically route a set of tracks through groups and subgroups in the mix. It uses openly available linked databases to infer the relationship between instrument objects in a DAW session, utilising graph theory and hierarchical clustering to obtain the groups. This can be used in any intelligent production environment to configure the sessions' routing parameters.

1. INTRODUCTION

Audio production is a complex task with many decisions needed to be undertaken by the engineers to achieve an end mix. If only gain coefficients exists, then a mix can be summarised as an n-dimensional vector, where n is the number of channels [1]. Introducing digital effects, panning and routing adds even more complexity to this model. Grouping allows the engineer to create *sub*-mixes, which can themselves be mixed further into the final mix space, effectively partitioning the mix-space.

This paper introduces a novel method to automatically group a set of given tracks based on their instrument metadata. The paper will give a brief overview of intelligent systems in section 2. The model will then be described in section 3, along with examples on each stage. The paper concludes with an evaluation of the tool in section 5.

2. BACKGROUND

Automated music production systems generally provide parameter recommendations from the audio signals [2,3]. These return environment control signals to the mix which map directly onto the parameters. Intelligent systems can instead recommend parameters or provide intuitive mappings, rather than directly control parameters [3, 4]. These tools aim to reduce the high dimensional problem of mixing into a lower number of dimensions. This is achieved by taking some contextual information, such as a semantic descriptor or user ratings, to build a low-dimensional interface for users to explore. This reduces the burden on engineers without removing the user control, as long as linearity is preserved.

Web-based knowledge stores are readily available to obtain semantic relationships. However instrument-based ontologies are not readily agreed upon [5], with individual instruments being linked to various categories based on creator preference. Therefore there is no exhaustive and peer-reviewed resource. Wikipedia holds a vast amount of information on instrument data which can be queried using a SPARQL end-point at DBPedia[1]. Each page on wikipedia have subject tags, which are subjects and categories containing that page. Each subject connects back to super-subjects and so forth. These are linked using the Simple Knowledge Organisation System (skos)[2] *broader* tags.

Graphs allow data to be represented as a structure and allow the relationship between vertices to be examined. These vertices can then be classified together by clustering based on distances to other vertices [6], or constructed based on distances to other data points [7].

Music production encompasses more than just mixing decisions, it also requires structuring a mix session. [8] shows sessions with more groupings per track tend to be perceived as better mixes, with higher perceptual ratings. It also shows tracks are grouped together because they are similar, with the group name representing the tracks included, such as 'vocals', 'Drums' or 'Guitars'. Tracks can also be grouped based on their acoustic similarity [8, 9] or based upon a semantic descriptor, such as the instrument and genre [10,11].

3. AUTOMATIC CHANNEL ROUTING

Automatically deriving a session structure using semantic labels requires knowledge of each instrument in the mix and their relationship to each other. Tracks can be tagged with metadata to identify their instrument. Tracks with the

[1] http://dbpedia.org/snorql
[2] https://www.w3.org/2004/02/skos/

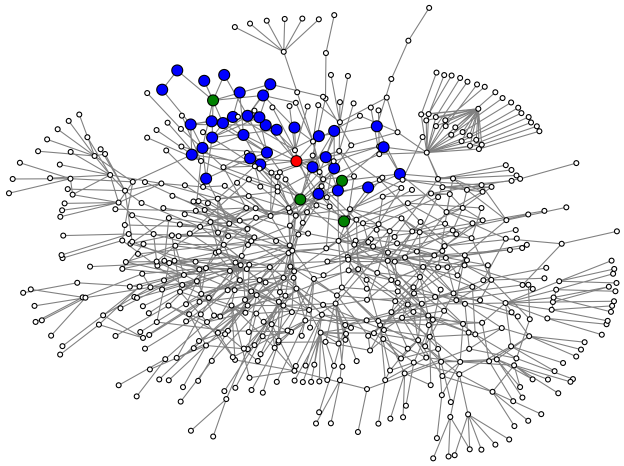

Figure 1: The full graph for four instruments: *Acoustic Guitar*, *Electric Guitar*, *Piano* and *Snare Drum*. The root *Musical instruments* is red, the instruments green and the nodes in the simple paths blue. This gives every possible subject which contains these four instruments to a depth of 4, showing the complexity of linked data stores. Cutting the graph gives a focused scope.

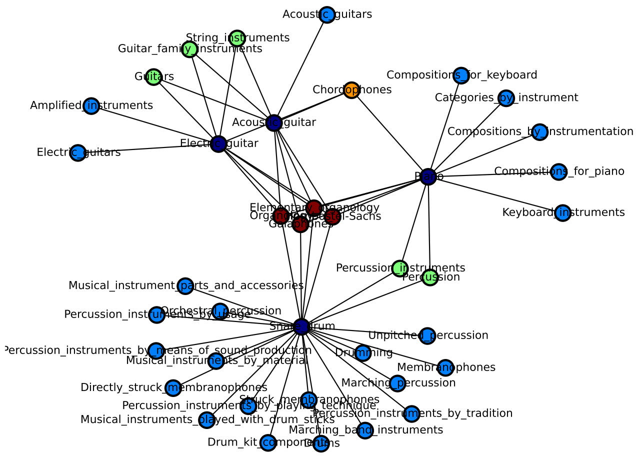

Figure 2: Forced neighbourhood graph G_2 from G_1, depicted in figure 1

same instrument are placed into a group together. This simplifies the process however a threshold on whether smaller groups should be discouraged could be implemented. A final list of the individual instruments is used as the input to the system. These instrument names map directly onto wiki pages, which can be queried using DBPedia. Once the page is known, the subjects which contain the page can be identified by traversing along the `skos:broader` tags. For example the ‘Piano’ is a member of the subject ‘String Instruments’, which itself is a member of ‘Musical Instruments’. These can be represented in a directional graph G, where the vertices are the pages (subjects and instruments) and the edges their relationships. The first problem is the number of nodes this creates; the ‘Piano’ node can link to over 310 subjects and 432 relationships when traversing just 4 levels. However, instruments should share a number of subjects between them.

Many subjects are captured which are not related to music and could distort the clustering process. The *Musical instruments* subject page is used as a subject root node (v_0) to help identify relevant information. The instrument vertices are referred to v_{inst} and must link back to the root v_0 vertex. If a subject is within the path between any instrument and the root subject, it is kept in the graph. Any vertices which do not match this criterion are discarded, giving the graph $G_1 \subset G$, $G_1 = (V_1, E_1)$. Figure 1 shows the full graph G for four instruments with the subgraph G_1 highlighted to show the extent of the cut.

3.1. Instrument similarity

The instruments and subjects are stored as vertices, therefore instrument similarity can be calculated using graph theory techniques. The similarity between two vertices can be evaluated by analysing the overlap in their neighbourhoods $\Gamma(v)$ using the Jaccard similarity coefficient [6], equation 1.

$$w(v,u) = \frac{|\Gamma(v) \cap \Gamma(u)|}{|\Gamma(v) \cup \Gamma(u)|} \tag{1}$$

Since G_1 is a relational graph, each instruments neighbourhood will be intentionally small, as each instrument may only be a direct ancestor of some subjects. Therefore, to get the best similarity score, each instrument vertex’s neighbourhood should be made of every subject vertex it has a path to in G_1. This flattened graph is called G_2 and has the same vertices as G_1, $V_2 = V_1$. Edge $\{v_{inst}, v_j\} \in E_2$ if a path between v_{inst} and v_j exists in G_1.

Now each instruments’ neighbourhood is comprised of every subject vertex it could connect to. Subjects which are common have a high number of connections, whilst specific subjects will connect to only a few instruments. However, instruments with similar subjects, and therefore similar neighbourhoods, should be themselves similar to each other. The flattened representation of figure 1 is given in figure 2 and shows the subject relationship to the four instruments. The subjects *Organology* and *Gaiaphones* are universally common to the four instruments and have been pushed centrally. Whilst the more specific subjects *Keyboard instruments* and *Amplified Instruments* are pushed outwards as they connect to one instrument only (*Piano* and *Electric Guitar* respectively).

The Jaccard similarity coefficient w can be calculated for every pair of instrument vertices. A coefficient $w = 1$ means identical neighbourhoods, and $w = 0$ means no

	Ac. Gtr	El. Gtr	Piano	Sn. Drum
Ac. Gtr.	0.000	0.294	0.750	0.875
El. Gtr.	0.294	0.000	0.783	0.886
Piano	0.750	0.783	0.000	0.793
Sn. Drm	0.875	0.886	0.793	0.000

Table 1: The distance matrix of the four instruments in figures 1 and 2.

commonality. This similarity measure can be converted into a distance measure by $1 - w$ and stored as an n-by-n matrix D, where n is the number of instrument vertices.

For the example set of four instruments in figures 1 and 2, the distance matrix is shown in table 1. It is clear that the matrix is symmetric since $w(v_i, v_j) = w(v_j, v_i)$ and $w(v_i, v_i) = 1$. The *Acoustic Guitar* and *Electric Guitar* are predictably very similar, with only 5 specific subjects from a union size of 17 subjects. The *Piano* and *Snare Drum* are expectedly not very similar to any other of the given instruments.

Hierarchical clustering is then performed on this distance measure, allowing for a distance relationship to be created. The result from the clustering can then be flattened into a set of discrete clusters. The number of clusters, k represents the number of distinct groups to create. From [11] a suitable measure for k can be $k = \min\left(\lfloor \frac{\mathrm{N}}{2} - 1 \rfloor, 1\right)$. Using the four instruments, the system recommended two groups: $C_0 = \{$*Acoustic Guitar*, *Electric Guitar*$\}$ and $C_1 = \{$*Piano*, *Snare Drum*$\}$.

3.2. Naming groups

The clusters make up the groups, with each instrument contained inside a member of the group. Traditionally groups are named and labelled within a DAW to represent the group, such as 'Drums' or 'Vocals'. The names can be found by identifying the nearest common subject to each of the instruments in the group. This should therefore be a subject which is representative of every instrument within that group.

To identify this subject, the G_1 graph is cut to only include the paths and nodes of the instruments inside the k-th group, such that $G_3^k \subset G_1$. Figure 3 shows this subgraph of G_1 for a cluster containing *Acoustic Guitar* and *Electric Guitar*. The distance between two vertices is the number of vertices needed to traverse to reach the target from the source. This is defined as $\delta(v_i, v_j)$. If v_i does not have a path to v_j then $\delta = \infty$.

The nearest common subject vertex for cluster C_k is defined as s_k and can be found using equation 2. Each vertex in G_3^k which is a subject is evaluated. The vertex with the smallest total distance from every instrument vertex v_{inst} in cluster C_k is the nearest comment vertex s_k. The group name is then given as the label attributed to this subject. In figure 3 this is the 'Guitars' subject and can be confirmed

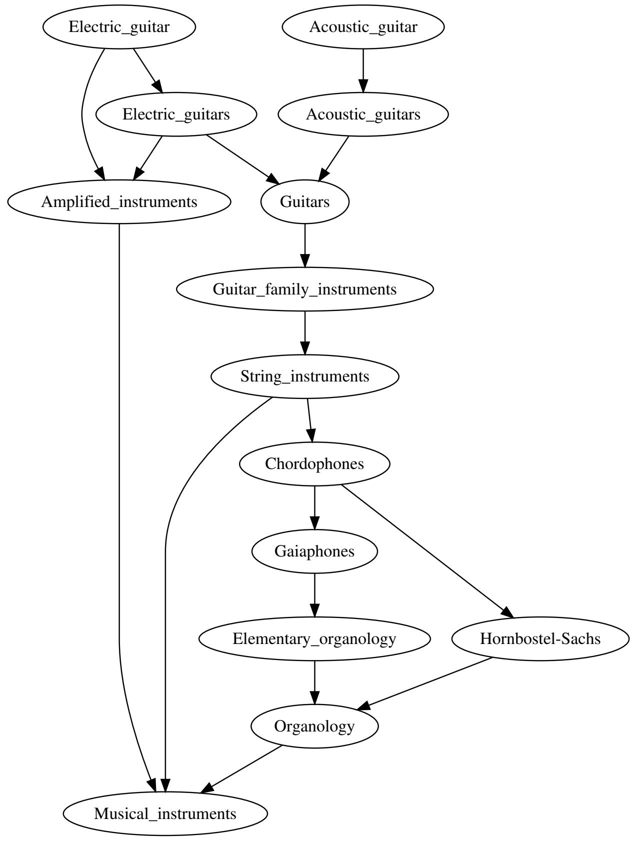

Figure 3: The cluster subgraph G_3^k formed from a cluster holding *Acoustic Guitar* and *Electric Guitar*

.

visually from the two instruments in the cluster *Acoustic Guitar* and *Electric Guitar*.

$$s_k = \underset{j}{\operatorname{argmin}} \left[\sum_{v_{inst} \in C_k} \left(\delta\left(v_{inst}, v_j\right)\right) \right] \tag{2}$$

An output of a 14 track input is depicted in figure 4. Tracks with the same instrument have been grouped together into *Electric Guitars*, *Snare Drum*, *Drum Kit* and *Tom-tom drum*. The final list of instruments to group are then processed to identify the final layer of groups, giving two supergroups. These groups then route to the master output.

4. CONCLUSION

This paper has presented a novel way of automating the practice of subgrouping in music production by utilising publicly maintained knowledge stores. The results shows that using such data can be useful so long as appropriate relationships are examined. These relationships can be analysed and interpreted using graph theory, specifically utilising techniques to measure the similarity between vertices.

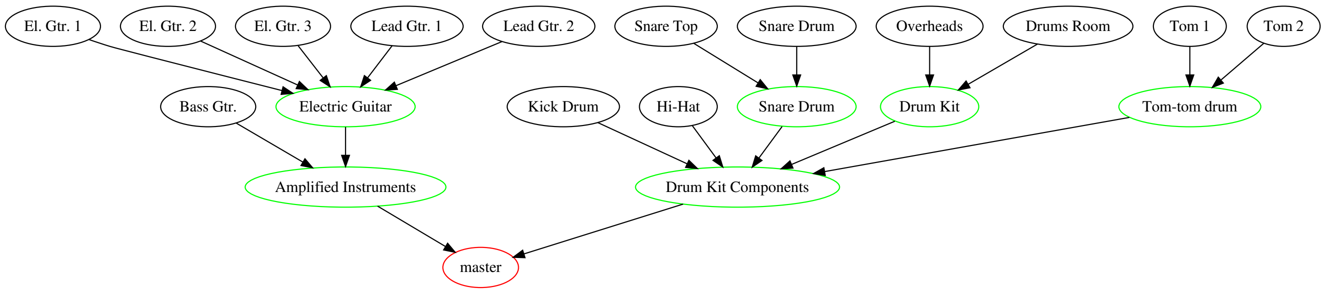

Figure 4: Complete output from a set of test tracks. 6 groups are recommended for the 14 tracks, judged only from their instrument labels.

For this case, vertex similarity leads to instrument similarity.

5. FURTHER WORK

Evaluating the performance is in itself a subjective problem. There does not exist a single, correct grouping of instruments available, only anecdotal results [10, 11]. However if a target or 'ideal' graph is identified then graph similarity techniques can be used.

Comparing two graphs together is not a trivial problem, with multiple metrics being available. A popular comparison metric is the size of the maximum common subgraph [12], tries to find the largest portion of two graphs which are isomorphic. In a tree, one false vertex may discard entire leafs, severely minimising the size of sub-graph that can be found.

The graph edit distance (GED) is a suitable measurement for comparing two graphs [12]. GED attempts to find the smallest number of edits required to convert G_A into G_B. Edits usually incorporate inserting and removing of edges and vertices. By attributing a cost to these actions, it is possible to evaluate how similar two graphs are. [12] gives suitable costs for each edit operation.

6. REFERENCES

[1] M. Terrell, A. Simpson, and M. Sandler, "The mathematics of mixing," *J. Audio Eng. Soc*, vol. 62, no. 1/2, pp. 4–13, 2014.

[2] S. Mansbridge, S. Finn, and J. D. Reiss, "Implementation and evaluation of autonomous multi-track fader control," in *Audio Engineering Society Convention 132*, Apr 2012.

[3] S. Stasis, R. Stables, and J. Hockman, "Semantically controlled adaptive equalisation in reduced dimensionality parameter space," *Applied Sciences*, vol. 6, no. 4, 2016.

[4] B. Pardo, D. Little, and D. Gergle, "Building a personalized audio equalizer interface with transfer learning and active learning," in *Proceedings of the Second International ACM Workshop on Music Information Retrieval with User-centered and Multimodal Strategies*, MIRUM '12, (New York, NY, USA), pp. 13–18, ACM, 2012.

[5] S. Kolozali, M. Barthet, G. Fazekas, and M. B. Sandler, "Knowledge representation issues in musical instrument ontology design.," in *ISMIR*, pp. 465–470, 2011.

[6] S. E. Schaeffer, "Graph clustering," *Computer Science Review*, vol. 1, no. 1, pp. 27 – 64, 2007.

[7] D. C. Corrêa, A. L. Levada, and L. d. F. Costa, "Finding community structure in music genres networks.," in *The 12th International Society for Music Information Retrieval Conference*, pp. 447–452, 2011.

[8] D. Ronan, B. De Man, H. Gunes, and J. D. Reiss, "The impact of subgrouping practices on the perception of multitrack music mixes," in *Audio Engineering Society Convention 139*, October 2015.

[9] D. Ronan, D. Moffat, H. Gunes, and J. D. Reiss, "Automatic subgrouping of multitrack audio," in *18th International Conference on Digital Audio Effects*, November 2015.

[10] N. Jillings and R. Stables, "Investigating music production using a semantically powered digital audio workstation in the browser," in *Audio Engineering Society Conference on Semantic Audio*, June 2017.

[11] D. Ronan, H. Gunes, and J. D. Reiss, "Analysis of the subgrouping practices of professional mix engineers," in *Audio Engineering Society Convention 142*, May 2017.

[12] H. Bunke, "On a relation between graph edit distance and maximum common subgraph," *Pattern Recognition Letters*, vol. 18, no. 8, pp. 689 – 694, 1997.