Proceedings of the 4th Workshop on Intelligent Music Production, Huddersfield, UK, 14 September 2018

TOWARDS A SEMANTIC WEB REPRESENTATION AND APPLICATION OF AUDIO MIXING RULES

David Moffat, Florian Thalmann and Mark B. Sandler

Centre for Digital Music
Queen Mary University of London
{d.j.moffat, f.thalmann, mark.sandler}@qmul.ac.uk

ABSTRACT

Existing literature has discussed the use of rule-based systems for intelligent mixing. These rules can either be explicitly defined by experts, learned from existing datasets, or a mixture of both. For such mixing rules to be transferable between different systems and shared online, we propose a representation using the Rule Interchange Format (RIF) commonly used on the Semantic Web. Systems with differing capabilities can use OWL reasoning on those mixing rule sets to determine subsets which they can handle appropriately. We demonstrate this by means of an example web-based tool which uses a logical constraint solver to apply the rules in real time to sets of audio tracks annotated with features.

1. INTRODUCTION

The field of intelligent audio production has existed, in some capacity, for almost 45 years [1]. One of the earliest approaches involved viewing automatic mixing as a constraint optimisation problem [2]. Since then, there have been several other examples of constraint optimisation for mixing [3–7]. Numerous papers have also discussed the use of a rule-based structure and format for automated mixing [8–11], though no rigid rule framework has ever been presented.

These different approaches to automatic mixing were implemented as separate systems with varying capabilities and differing formats were used to represent the rules. The aim of this paper is to investigate how semantic web technologies, in particular the Rule Interchange Format (RIF), can be used to represent audio mixing rules so that they can be shared across systems with different purposes. We illustrate how this can be done by implementing a set of example rules and using them across two systems with different purposes, a production-side automatic mixing plugin and a consumer-side adaptive listening tool.

2. THE SEMANTIC WEB AND THE RULE INTERCHANGE FORMAT

The Semantic Web is an extension of the World Wide Web that uses knowledge representation techniques to annotate web documents with meaning as opposed to unstructured plain text, so that they can be read and processed by computers. Real-world concepts and the relationships between them are modeled as ontologies, which can refer to each other and are typically shared between many different applications. There are many ontologies for representing various kinds of musical data, including musical works [12], the studio production process [11], or automatic audio analysis [13]. Data represented using such ontologies can then be aggregated into large graph structures, queried and reasoned upon automatically, based on a given set of rules. There are many existing Semantic Web rule languages, each of which have their own characteristics and limitations.

In order to mediate between these different rule languages, the Rule Interchange Format (RIF)[1] was introduced as a standardised framework for the application-independent representation of logical rules, which include both declarative and production rules. It consists of a set of interconnected rule dialects, which are represented as extensions of one and the same core representation RIF-Core[2]. The advantage of a standardised rule framework is that different systems, across multiple platforms, are able to exchange rule sets and adapt them for their own purpose. This allows for differing pieces of content to have the same rulesets applied even when using distinct dialects and reasoning engines, without the need for bespoke design and implementation of rule structures and reasoning engines. Any new application can embed an appropriate reasoner, and reason over the given logical ruleset.

3. MIXING RULE GENERATION

Creating mixing rules can be a complex task. There are two common approaches to rule generation, which are often also combined. The first approach consists in gathering expert knowledge from the literature, e.g. mixing process descriptions or idiosyncratic tricks, and manually translating the knowledge into specific rules. Such an approach is discussed in [8]. A second approach involves automatically learning rules from a given dataset, which can consist of separate stems (the input material) and audio mix-downs at different stages of the mix, or even recordings of parameter

[1] https://www.w3.org/TR/rif-overview/
[2] https://www.w3.org/TR/2013/REC-rif-core-20130205/

values during the mixing process, as demonstrated in [7].

In the context of this work we created a small set of rules taken from the literature, as a simple starting point to test the feasibility of the approach.

For rules to be exchangeable between different automatic mixing systems, the systems must implement a comparable model. Generally, an automatic mixing system takes a number of audio tracks as an input and yields a single mono or stereo track as an output.

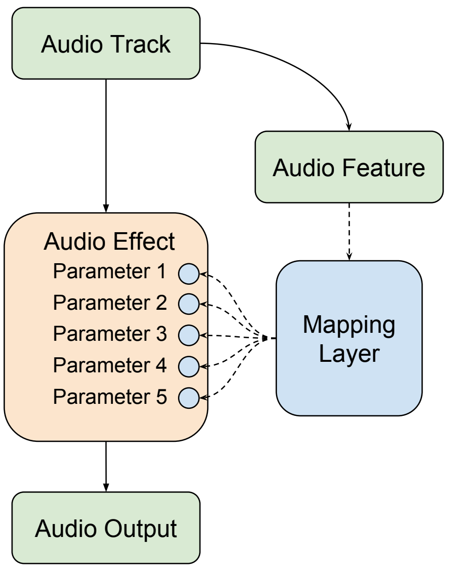

Figure 1: Flow diagram of a basic automated mixing production system. A solid line is the transfer of an audio signal, and a dotted line is the transfer of meta-data

In our experiment, we consider a simple, adaptive audio effect, where all input audio tracks can be considered separate from each other, as demonstrated for a single track in Figure 1. The incorporation of all other audio features from all other tracks is considered a component of the mapping layer.

An audio track is taken, as defined within the music ontology [12]. From these a set of audio features can be extracted [14], and represented using the Audio feature ontology [13]. This audio feature representation is then passed to a mapping layer. This mapping layer rescales and represents the mixing control parameters of audio effects, typically from some high-level representation [11], including providing the cross-adaptive control of audio tracks. This mapping later will then enact some control parameters of an associated audio effect, as described by the audio effect ontology [15]. From this given structure, RIF rules can be produced in any of the three different ways discussed above.

4. AN EXAMPLE MIXING RULE IN RIF

A rule set can be constructed using the RIF rule framework as presented in Listing 1. The example set consists of a list of rules which sets a parameter to a specified value under a given condition.

The `if` checks conditions on three variables and selects anything as an audio track that has a loudness features and a gain audio parameter associated with it. The `then` specifies the gain parameter value to be set under the condition, in dB, such that all tracks have a loudness of $-12dB$.

Listing 1: Example mixing rule in BLG Presentation Syntax

```
Document(
  Base( <http://www.w3.org/2007/rif#>)
  Prefix(prd <http://www.w3.org/2007/rif#>)
  Prefix(dymo <http://tiny.cc/dymo-ontology#>)
  Prefix(aufo <https://w3id.org/aufx/ontology
      /1.0#>)
  Prefix(afo <https://w3id.org/afo/onto/1.1#>)
  Prefix(func <http://www.w3.org/2007/rif-builtin
      -function#>)
  Prefix(pred <http://www.w3.org/2007/rif-builtin
      -predicate#>)

  Group(
    Forall ?track ?loudness ?gain(
      If
        And(
          :type(?track dymo:Dymo)
          :hasFeature(?track ?loudness)
          :type(?loudness afo:Loudness)
          :type(?gain aufo:Gain)
          :hasParameter(?track ?gain)
        )
      Then
        Modify(
          :value(?makeupGain, func:subtract(-12,:
              value(?loudness)))
          External(:value(?gain, ?makeupGain))
        )
    )
  )
)
```

5. APPLICATION OF SEMANTIC MIXING RULES

The RIF rules obtained as described can then be transferred as Semantic Web documents, e.g. represented in RDF, to specific applications that know how to apply such rules. For illustration here, we do this in two different contexts, studio production and consumer listening experience.

In order to import a given set of RIF rules into a system one typically goes through several steps:

- **filtering**: The rules first need to be filtered to find the subset that the current system can solve. Since the rules themselves are represented in OWL, this can be done using a common Semantic Web reasoner, e.g. an OWL reasoner. Each system defines its preselection rules, such as a rule that ignores any RIF rule using arithmetics.

- **mapping**: Then, the rules possibly need to be converted into a format that the system understands. If the system uses any other Semantic Web format, this can be done with a system-specific ontology mapping. Otherwise, they need to be serialized to the system's rule language using for instance an RDF parser or SPARQL queries.

- **application**: Finally, the rules can be applied to a given set of tracks using the system's own reasoner.

5.1. Mixing Studio Plugins

In the context of a recording studio, rules can be both learned and applied. On the one hand, during the mixing process using a Digital Audio Workstation (DAW), such as a Web DAW [16], a analytical plugin can be used to record and analyse the application of audio effects to sets of stems and generate a given RIF mixing rule file, which can then be shared among other projects or engineers. On the other hand, an automatic mixing plugin can, upon importing such mixing rules, apply them to a given set of audio stems through logical constraint satisfaction using a suitable built-in reasoner. Example situations where engineers could benefit from the exchange of such rule sets, are the application of standard initial settings typical for a particular engineer, the generation of a rough mix, or the exploration of the effect of different mixing approaches.

5.2. Contextual Listening

Similarly, such mixing rules collected in the studio may be useful for achieving contextual listening in consumer-oriented applications. An example of such an application is the Semantic Player [17], a web and mobile app based on the Web Audio API which explores new ways of playing back music in indeterminate, context-dependent, and interactive ways. It is based on the Dynamic Music Objects format, which represents musical content as multi-hierarchical structures and makes it modifiable within definable constraints. For each Dynamic Music Object, the Semantic Music Player generates a custom graphical interface and allocates any used sensors and data sources.

The latest version of the Dynamic Music Objects framework[3] includes an in-browser Semantic Web triple store which holds musical structures at runtime and which can be queried and reasoned upon. Due to the limitations of current Semantic Web reasoning engines for real-time and stream-based data, the framework includes a constraint satisfaction reasoner based on LogicJS[4] and defines its own representation format based on syntax trees of logical expressions[5], comparable to RIF. The current expressivity of the rule system allows for equalities and inequalities, arithmetic operations, as well as arbitrary code fragments, which are executed as javascript functions.

For the example rule in Listing 1, the system generates one constraint for each of the available tracks, each of which will simply be applied upon initialisation to set its associated track's `aufo:Gain` parameter accordingly, and since the feature `afo:Loudness` is unmodifiable, the system reaches it's final state. This rule can be applied to any set or hierarchy of Dynamic Music Objects that have loudness features and a gain parameters. However, in practice, with larger hierarchies one might also need to constrain the hierarchical level on which the constraint applies, simply by adding e.g. `:hasFeature(?track ?level)` and `:type(?level dymo:Level) :value(?level x)` to the `And` clause, where `x` is the level in question.

In order to import RIF rules into the Semantic Player, we define an ontology mapping from RIF to the player's own expression ontology and convert the set of rules to ones represented using the expression ontology. Then, these rules can be added to the rendering definition of any Dynamic Music Object that consists of a simple hierarchy of tracks.

6. CONCLUSION

We have discussed the principle of using a rigid form of mixing rules, and demonstrated an approach towards the development of a fixed rule framework for consistent transportation and interpretation of such rule sets. Semantic technologies provide a standard and easy to interpret platform for rule sets and can provide an approach to constraint optimisation based on these rules. It has been shown that adaptive rule can be applied in a simple way and interpreted with a logical reasoner. It has also been shown that with simple definitions, these rules can be applied to hierarchical musical structures [18], or more complex musical structures [17], with no significant changes to either the rule framework or implementation platform.

7. REFERENCES

[1] D. Dugan, "Automatic microphone mixing," *Journal of the Audio Engineering Society*, vol. 23, no. 6, pp. 442–449, 1975.

[2] F. Pachet and O. Delerue, "On-the-fly multi-track mixing," in *Audio Engineering Society Convention 109*, Audio Engineering Society, 2000.

[3] B. Kolasinski, "A framework for automatic mixing using timbral similarity measures and genetic optimization," in *Audio Engineering Society Convention 124*, Audio Engineering Society, 2008.

[4] D. Barchiesi and J. Reiss, "Automatic target mixing using least-squares optimization of gains and equal-

[3]`https://github.com/dynamic-music/dymo-core`
[4]`https://github.com/mcsoto/LogicJS`
[5]`https://tiny.cc/expression-ontology`

ization settings," in *Proceedings of the 12th Conference on Digital Audio Effects (DAFx-09), Como, Italy*, pp. 7–14, 2009.

[5] M. J. Terrell and J. D. Reiss, "Automatic monitor mixing for live musical performance," *Journal of the Audio Engineering Society*, vol. 57, no. 11, pp. 927–936, 2009.

[6] M. Terrell, A. Simpson, and M. Sandler, "The mathematics of mixing," *Journal of the audio engineering society*, vol. 62, no. 1/2, pp. 4–13, 2014.

[7] A. Wilson and B. M. Fazenda, "An evolutionary computation approach to intelligent music production informed by experimentally gathered domain knowledge," in *Proceedings of the 2nd AES Workshop on Intelligent Music Production*, 2016.

[8] B. De Man and J. D. Reiss, "A knowledge-engineered autonomous mixing system," in *Audio Engineering Society Convention 135*, Audio Engineering Society, 2013.

[9] A. L. Benito and J. D. Reiss, "Intelligent multitrack reverberation based on hinge-loss markov random fields," in *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*, Audio Engineering Society, 2017.

[10] P. Pestana and J. Reiss, "Intelligent audio production strategies informed by best practices," in *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*, Audio Engineering Society, 2014.

[11] T. Wilmering, G. Fazekas, and M. B. Sandler, "High-level semantic metadata for the control of multitrack adaptive digital audio effects," in *Audio Engineering Society Convention 133*, Audio Engineering Society, 2012.

[12] Y. Raimond, S. Abdallah, M. B. Sandler, and F. Giasson, "The music ontology," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2007.

[13] A. Allik, G. Fazekas, and M. Sandler, "An ontology for audio features," in *17th International Society for Music Information Retrieval Conference*, pp. 258–275, 2016.

[14] D. Moffat, D. Ronan, and J. D. Reiss, "An evaluation of audio feature extraction toolboxes," in *Proc. 18th International Conference on Digital Audio Effects (DAFx-15)*, November 2015.

[15] T. Wilmering, G. Fazekas, A. Allik, and M. B. Sandler, "Audio effects data on the semantic web," in *Audio Engineering Society Convention 139*, Audio Engineering Society, 2015.

[16] N. Jillings and R. Stables, "An intelligent audio workstation in the browser," in *3rd Web Audio Conference (WAC)*, (London, UK), August 2017.

[17] F. Thalmann, A. Perez-Carrillo, G. Fazekas, G. A. Wiggins, and M. Sandler, "The semantic music player: A smart mobile player based on ontological structures and analytical feature metadata," in *2nd Web Audio Conference (WAC)*, (Atlanta, Georgia, USA), 2016.

[18] D. Ronan, D. Moffat, H. Gunes, and J. D. Reiss, "Automatic subgrouping of multitrack audio," in *Proc. 18th International Conference on Digital Audio Effects (DAFx-15)*, DAFx-15, November 2015.