likely music

Probabilistische Musiknotation

Lukas Epple

24. September 2017

Zusammenfassung

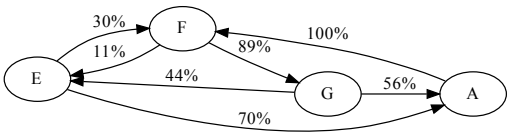
likely music ist eine Software, um probabilistiche Musik zu notieren und abzupsielen. Probabilistische Musik heißt in diesem Falle, dass die Interpretation der vorliegenden Notation deutlich freier ist als bei herkömmlicher Musik und auch die Reihenfolge der Noten betrifft. Um dies zu erreichen wird ein eigenes Modell von Musiknotation verwendet. An Stelle der Lineare Reihenfolge von Noten bzw. Akkorden tritt ein Graph, in dem die Noten (bzw. Akkorde) die Knoten und die Kanten die möglichen Übergange zwischen diesen darstellen, wobei jede Kante eine gewisse Wahrscheinlichkeit zugeordnet ist. Dieses Modell ist unter anderem sehr gut von einem Computer zu fassen, wodurch es möglich wird, solche Notationen automatisch zu „interpretieren“ bzw. abzuspielen, indem eine Notenabfolge gemäß der Notation ausgewürfelt wird.

likely music kann also sowohl probabilistische Noten erstellen und editieren, als auch mittels MIDI diese abspielen oder als Audiodateien exportieren.

Idee

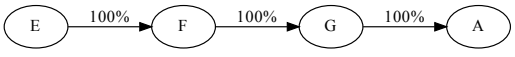
Der eigentlichen Idee ging ein mehr oder minder gescheitertes Projekt für diesen Wettbewerb vorraus. Im Frühjahr diesen Jahres entschied ich mich dieses, eine Demo [1], abzubrechen, einfach weil ich befürchtete, es nicht bis zur Frist fertigstellen zu können. Die Motivation für dieses Projekt speiste sich aus meiner Faszination für Demos an sich, denn ich hatte bereits im Vorfeld öfters mich mit diesen beschäftigt und beim Ansehen der Einsendung von Demo-Wettbewerben ein Bedürfnis entwickelt auch so etwas zu entwickeln. Das neue Projekt speiste sich aus einer weiteren Faszination von mir, nämlich einer für Kunst, die basierend auf Kunst entsteht. Ich erinnere mich oft besonders an Kunstinstallationen, die ihr gestaltendes Element durch Zufall oder einen undurchschaubaren oder chaotischen Prozess bezieht. Beim Nachdenken über Zwölftonmusik, die – meiner Meinung nach – ein wenig jenen Elements hat, kam mir die Grundidee – wie ich mich erinnere – auf dem Gang zwischen zwei Schulstunden für *likely music*, nämlich ein Modell, um Musik zu beschreiben, die zufällig im Vortrag ist.

Das Modell, das ich übertrieben panisch auf ein Stück Notizblock kritzelte, sieht Musik als gerichteten Graphen, wobei die Knoten Musiknoten einer bestimmten Länge und die Kanten zwischen ihnen die Wahrscheinlichkeit des Wechsel von der einen Note zu anderen. Vorstellen kann man sich es in etwa so:



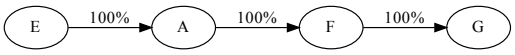
In diesem konkreten Graphen sind die Noten E, F, G und A als Knoten vertreten (der Einfachheit halber sind die Notenlängen weggelassen). Beispielsweise vom E führen zwei Kanten weg, eine zum F mit dreißigprozentiger Wahrscheinlichkeit und eine zum A mit siebzigprozentiger Wahrscheinlichkeit, d. h. nach dem E kommt in sieben von zehn Fällen das A und in den drei übrigen das F, analog gilt verhält es sich mit den anderen Noten.

Diese Darstellung ist in gewisser Weise auch nur eine ausdrucksstärkere Form einer normalen Notation, denn ein Weg durch den obigen Graphes könnte so aussehen:



Diese Interpretation, die eine Wahrscheinlichkeit von ca. 15% hat aufzutreten, entspricht einer einfachen, linearen Notation wie sie in einem Gesangsbuch stehen könnte. Wir sehen also, dass solche

probabilistiche Noten (wie unser Graph von vorhin) durch ein Verfahren, das ich einfach in einer Erweiterung des Begriffs als Interpretieren bezeichen, auf eine lineare Notation reduziert werden kann, die mit einem Instrument oder vom Computer gespielt werden kann. Es ist sogar nicht nur eine lineare Notation, sondern – je nach vorgegebenen Graph – eine Vielzahl ihrer möglich. Beispielsweise wäre eine weitere:



Ähnlich gibt es noch viele weiter Möglichkeiten. Zu beachten ist bei den beiden Beispielinterpretationen noch: Sie sind nach vier Noten abgeschnitten, denn, da von jedem Knoten mindestens eine Kante ausgeht, könnte man den Graphen potentiell unendlich lang ablaufen und würde somit eine unendlich lange Interpretation generieren.

Was aus dieser Grundidee zu machen war, schien mir von Anfang an recht klar: Als Software implementieren, um ein graphisches Interface bereitzustellen, das es erlaubt, probabilistische Notation zu erstellen, zu editieren und abzuspielen.

Umsetzung

Gleich zu Beginn war klar, dass Haskell die Programmiersprache der Wahl werden sollte. Sie ist die Sprache, die ich in den letzten Jahren am aktivsten verwendet habe und mir einiges bietet, statische Typisierung, um Fehler vorzubeugen, ein expressives Typsystem, das es erlaubt, Daten besser zu strukturieren, und funktionale Programmierparadigmen, die mir persönlich sehr gut taugen, um mal einige zu nennen.

Zunächst konzentrierte ich mich darauf, den Graphen und den Interpretationsalgorithmus als Bibliothek zu implementieren. In der ersten Iteration dieser Bibliothek, noch *probable music* genannt, begann ich auch einen eigenen Softwaresynthesizer zu implementieren, der flexibel auf verschiedenen Plattformen und zu verschiedenen Zwecken verwendet werden kann. Der Synthesizer konnte – gegeben ein Algorithmus dafür – jegliche Daten in Töne umwandeln, was interessante Möglichkeiten ergab, sich außerhalb des Zwölftonsystems zu bewegen. Die Tonerzeugung basierte dann auf einer freien Monade [2], die die Instruktionen ›Warten‹ und ›Abspielen‹ kannte. Indem man diese Instruktionen für verschiedene Audiosystem, wie SDL [4], Jack [3] oder auch Audiodateien wie WAV [5], implementierte, konnte man verschiedene Plattformen unterstützen. Allerdings gestaltete es sich schwierig, einen gut klingenden Synthesizer zu schreiben schwierig, denn die Messlatte ist im Vergleich zu realen Instrumenten hoch. Hinzu kamen noch einige Performance-Probleme mit meinem macschinennahem Audio-Code.

Also entschied ich mich, die Library vor allem auf den Graphen und die dazugehörigen Algorithmen zu fokusieren und zur Tonerzeugung eine geeignete Abstraktion zu verwenden, die diese zu vereinfachern. Ich habe hierfür MIDI gewählt, eine Technologie, die schon lang in allen Arten von Software und Hardware zur Musikproduktion verwendet wird, entschieden. MIDI basiert auf einer Abfolge von zeitlich abgestimmten Nachrichten, wie zum Beispiel ›Note C an‹ oder ›Note C aus‹. Aufgrund dieser Nachrichten kann man die Erzeugung und das Abspielen von Musik zwischen mehreren Programmen aufteilen, außerdem erlaubt es die bereits existierende Infrastruktur für MIDI-Verarbeitung zu verwenden, die sehr beachtlich ist. Für MIDI verwendet *likely music* die Open-Source-Bibliothek Euterpea[1] [8], die unter anderem eine kleine Abstraktion über MIDI enthält. Sie erlaubt es, in einem internen Format Musik zu konstruieren und anschließend als MIDI zu exportieren bzw. an ein anderes Programm zur Weiterverarbeitung zu schicken.

Bei der Darstellung des Graphen habe ich mich vor allem darauf konzentriert, dass der Interpretationsalgorithmus, also das (zufällige) Ablaufen des Graphen, möglichst effizient zu machen. Da es sich um einen gerichten Graphen handelt, ist es besonders wichtig zu wissen, wohin man von einem gegebenen Knoten aus gelangen kann bzw. welche Kanten von einem Knoten weggehen. So gelangt man in unserem Beispiel aus dem vorherigen Kapitel vom Knoten mit dem E zu den Knoten mit F und A. Es

[1] Ich musste allerdings aufgrund von Inkompatibilitäten mit den aktuellen Haskell-Paketen diese selbst beheben [9]. Diese Änderung wartet [10] aktuell (Stand 23.09.2017) darauf vom Hauptentwickler in den Code von Euterpea übernommen zu werden.

muss also möglichst effizient sein, die Kanten nachzuschlagen, die von einem Knoten *wegführen*. Mit der Datenstruktur *Map* [11] (im deutschen Sprachgebrauch typischerweise *assoziative Datenfeld* bzw. *assoziatives Array*) kann man genau das sehr leicht realisieren, indem man die Knoten als Schlüssel und eine Liste von Kanten, die vom Schlüssel weggehen, als Elemente verwendet. Wenn der Algorithmus nun einen Knoten nachschlägt, erhält er direkt die Kanten, die von diesem Knoten weggehen und somit auch die nächsten möglichen Knoten. Dies ist die einzige Information, die in jedem Schritt benötigt wird. Die Operation des Nachschlagen hat in einem *Map* die Komplexität $O(\log n)$ [12], d. h. die Zeit, die benötigt wird, um ein Element nachzuschlagen, steigt mit dem Wachsen der Datenstruktur logarithmisch (d. h. weniger starkes Wachstum als linear!), wodurch auch das Interpretieren großer Graphen ziemlich schnell bleibt. Der Code für die Datenstruktur findet sich im Abschnitt Library, Zeile 30 bis 43.

Der Interpretationsalgorithmus selbst ist rekursiv [15] gestaltet und findet sich in der Funktion `interpretation`, siehe Abschnitt Library, Zeile 52 bis 60. Diese Funktion benötigt einen intialisierten Pseudozufallszahlengenerator [13, 14], den zu interpretierenden Graphen in der eben besprochenen Datenstruktur und einen Startknoten und gibt die resultierende Interpretation im MIDI-Format von Euterpea [8] zurück. Zunächst wird der Startknoten im Graphen nachgeschlagen, so werden die Kanten bzw. die nächsten möglichen Knoten erhalten. Nun gibt es zwei Möglichkeiten für den weiteren Verlauf:

1. Es gibt keine Kanten, die von diesem Knoten ausgehen. Also wird die bisher generierte Interpretation einfach zurückgegeben, die Funktion terminiert.

2. Wenn es eine oder mehr Kanten vom Knoten aus gibt, wird eine (reelle) Zufallszahl zwischen 0 und 1 berechnet und mittels der Hilfsfunktion `edgeForRoll` (siehe Abschnitt Library, Zeile 62 - 67) die Kante erhalten, die gemäß des zufälligen Ergebnis als nächstes abgelaufen werden soll. Nun ergibt sich das gleiche Problem wie zu Beginn der Interpretation: Man kennt einen Knoten und will wissen wie es weitergeht. Also wird nach der Ermittlung des zweiten Knotens die MIDI-Nachrichten aus dem Startknoten extrahiert und dann der Interpretationsalgorithmus nochmal bzw. rekursiv aufgerufen – nur mit dem Folgeknoten als Startknoten – dessen Ergebnis wird an die aktuellen MIDI-Nachrichten angehängt, was jener Aufruf auch seinerseits wieder macht. So entsteht rekursiv eine (potentiell unendliche) Verkettung von MIDI-Nachrichten, die letztlich die finale Interpretation ergeben.

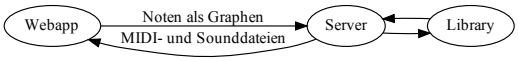
Da die meisten Graphen vermutlich vollständig untereinander verbunden sein werden wie zum Beispiel der Beispielgraph im ersten Abschnitt, entstehen unendlich lange Interpretationen. Diese zu erstellen benötigt natürgemäß natürlich auch unendlich viel Zeit – der Interpretationsalgorithmus terminiert also nicht. Die einfache Antwort auf dieses Problem ist die Begrenzung der Länge der Interpretation auf eine gewisse Anzahl von Noten, was sich dank eines Sprachfeatures von Haskell – Lazy Evaluation [16] – leicht umsetzen lässt. Denn mit Lazy Evaluation wird nur das berechnet, was im Moment benötigt wird. Somit werden zum Beispiel nur die ersten vier benötigten Noten berechnet und nicht die unendlich vielen die eigentlich noch darauf folgen würden – genau dies wird durch die Funktion `takeNotes` (siehe Abschnitt Library, Zeile 79 - 86) realisiert.

Nun können wir probabilistische Musik in Graphen darstellen, diese automatisch interpretieren und dank Euterpea nach MIDI exportieren. Was fehlt, ist eine angenehme Benutzerschnittstelle.

Zur Technologie für die Benutzerschnittstelle gab es für mich folgende Überlegungen: Zum einen sollte es leicht portabel bzw. auf jedem System laufen sowie außerdem einen begrenzten Entwicklungsaufwand mit sich bringen, sodass es bis zur Abgabe auch fertig sein würde. Ich selbst entwickle meine Software auf GNU/Linux, aber zur Abgabe müsste es auf macOS und / oder Windows laufen. Alle größeren Frameworks für Graphische Interfaces für GNU/Linux, wie zum Beispiel Qt [21] oder GTK [22], laufen auch auf den anderen großer Betriebssystemen. Allerdings bin ich nicht besonders vertraut mit irgendeinem dieser Frameworks, außerdem war ich mir nicht sicher, wie stressfrei die Verwendung dieser von Haskell aus sein würde (denn klassischerweise verwendet man C oder C++). Also entschied ich mich *likely music* als Webapplikation, die einfach in gängigen Browsern läuft zu

implementieren. Das hat einige Vorteile für mich, unter anderem, dass es leicht zu testen ist, weil die Browser eigentlich überall gleich sind, und, dass ich schon einige Erfahrung in Webentwicklung hatte.

Allerdings hatte ich die Library schon in Haskell implementiert, in Browsern läuft aber nur JavaScript (ohne größeren Aufwand zumindest). Also musste also ein Zwischenstück her um die Kommunikation zwischen der Library und der Webapplikation zu realisieren. Ich entschied mich für eine Client-Server-Architektur [17], also einen Server, der die Interpretation und den Export von Sounddateien für den Client, also die Webapplikation, übernimmt. Der Client wiederum müsste sich ausschließlich um ein ansprechendes Interface kümmern. Die ungefähre Gesamtarchitektur sieht also nun so aus:



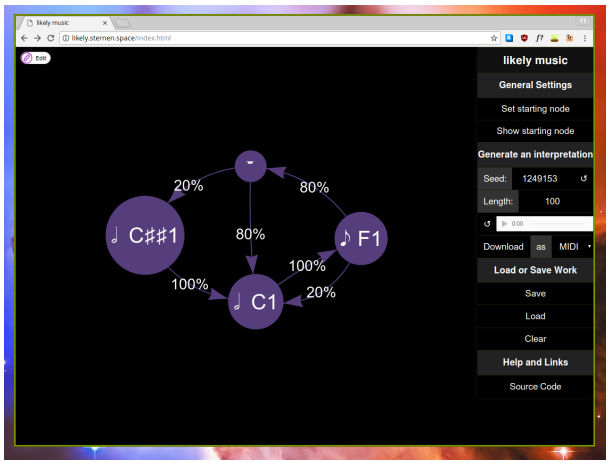
Der Server basiert auf den Libraries servant [18] als Webframework. Wie im Abschnitt **??** zu sehen, besteht das Serverbackend aus zwei Dateien Quelltext: In `Api.hs` wird die Struktur der REST-API [19] definiert, mittels der die Webapplikation mit dem Server kommuniziert. Der Server bietet folgende Funktionalität an:

- `/interpretation/mid` An diesen Endpunkt schickt die Webapplikation einen Graphen plus einiger Parameter in Form von JSON [20] und erhält eine Interpretation auf Basis des Algorithmus als MIDI-Datei zurück.
- `/interpretation/wav` Gleich wie der obige Endpunkt, allerdings wird vorher noch das MIDI mittels eines MIDI-Synthesizers, fluidsynth [**?**], in eine WAV-Datei konvertiert, sodass man es direkt anhören kann.
- Außerdem liefert er die statischen Dateien der Webapplikation, wie das nötige HTML, JavaScript und CSS.

Die erwähnten Parameter sind nur folgende drei:

- Der Anfangsknoten der Interpretation im Graphen, den der Algorithmus benötigt (wie oben besprochen).
- Die Länge der Interpretation als die maximale Anzahl an Noten in der Interpretation.
- Der Startwert für den Pseudozufallszahlengenerator [14], der für die Interpretation verwendet werden soll. Da derselbe Startwert in die selbe Interpretation resultiert, erlaubt dies sich interessante Interpretationen zu merken und zum Beispiel zu einer Interpretation noch die MIDI-Version zusätzlich herunterzuladen.

Dies ist auch schon alles, was das Serverbackend tut, denn es ist nur als minimaler Aufsatz auf die Library konzipiert. Das meiste für Benutzer relevante passiert in der Webapplikation, die folgendermaßen aussieht:



Lizenzierung

Benutzung

Zukünftige Weiterentwicklung

Links

- Der gesamte Quelltext `https://github.com/sternenseemann/likely-music`
- Eine laufende Instanz von *likely music* `https://likely.sternen.space`

Literatur

[1] `https://de.wikipedia.org/wiki/Demoszene`

[2] http://www.haskellforall.com/2012/07/purify-code-using-free-monads.html

[3] http://www.jackaudio.org/

[4] https://www.libsdl.org/index.php

[5] https://de.wikipedia.org/wiki/RIFF_WAVE

[6] https://www.midi.org/

[7] https://de.wikipedia.org/wiki/Musical_Instrument_Digital_Interface

[8] https://hackage.haskell.org/package/Euterpea

[9] https://github.com/sternenseemann/Euterpea2

[10] https://github.com/Euterpea/Euterpea2/issues/16

[11] https://hackage.haskell.org/package/containers-0.5.10.2/docs/Data-Map-Lazy.html#t:Map

[12] https://hackage.haskell.org/package/containers-0.5.10.2/docs/Data-Map-Lazy.html#v:lookup

[13] https://hackage.haskell.org/package/random-1.1/docs/System-Random.html#t:RandomGen

[14] https://en.wikipedia.org/wiki/Pseudorandom_number_generator

[15] https://de.wikipedia.org/wiki/Rekursion

[16] https://de.wikipedia.org/wiki/Lazy_Evaluation

[17] https://en.wikipedia.org/wiki/Client%E2%80%93server_model

[18] https://hackage.haskell.org/package/servant

[19] https://de.wikipedia.org/wiki/Representational_State_Transfer

[20] http://json.org/

[21] https://www.qt.io/

[22] https://www.gtk.org/

Anhang

Quelltext

Library

lib/Sound/Likely.hs

```
1  {-# LANGUAGE OverloadedStrings #-}
2  {-# LANGUAGE FlexibleInstances #-}
3  module Sound.Likely
4    ( Probability
5    , ID
6    , Node (..)
7    , Edge (..)
8    , Graph (..)
9    , insertNode
10   , insertEdge
11   , interpretation
12   , takeNotes
13   , emptyMusic
14   , exampleGraph
15   ) where
16
17 import Control.Monad
18 import Data.Aeson
19 import Data.Aeson.Types (Parser ())
20 import Data.Maybe
21 import Data.Text (Text ())
22 import Euterpea
23 import System.Random
24 import qualified Data.Map as M
25 import qualified Data.Set as S
26
27 type Probability = Double
28 type ID = Text
29
30 data Node
31   = Node
32   { nId :: ID
33   , nMusic :: Music Pitch
34   } deriving (Show, Eq, Ord)
35
36 data Edge
37   = Edge
38   { eTo    :: Node
39   , eProb :: Probability
40   } deriving (Show, Eq, Ord)
41
42 newtype Graph = Graph { unGraph :: M.Map Node (S.Set Edge) }
43   deriving (Show, Eq, Ord)
```

```
44
45 insertNode :: Node -> Graph -> Graph
46 insertNode t = Graph . M.insertWith S.union t S.empty . unGraph
47
48 insertEdge :: Node -> Edge -> Graph -> Graph
49 insertEdge n e =
50   insertNode n . Graph . M.insertWith S.union n (S.singleton e) . unGraph
51
52 interpretation :: RandomGen g => g -> Graph -> Node -> Music Pitch
53 interpretation gen graph n = (nMusic n) :+:
54   recurse (fromMaybe S.empty (M.lookup n (unGraph graph)))
55   where (prob, gen') = randomR (0.0, 1.0) gen
56         recurse edges =
57           if S.null edges
58             then emptyMusic
59             else interpretation gen' graph
60                  . eTo . edgeForRoll prob $ edges
61
62 edgeForRoll :: Probability -> S.Set Edge -> Edge
63 edgeForRoll prob set =
64   let curr = S.elemAt 0 set
65     in if prob <= eProb curr
66          then curr
67          else edgeForRoll (prob - eProb curr) (S.delete curr set)
68
69 emptyMusic :: Music a
70 emptyMusic = Prim (Rest 0)
71
72 exampleGraph :: Graph
73 exampleGraph = Graph $ M.fromList
74   [ (Node "bla" (c 4 qn), S.fromList [ Edge (Node "blub" (d 4 qn)) 1 ] )
75   , (Node "blub" (d 4 qn), S.fromList [ ])
76   ]
77
78 -- | Take the first @n@ notes of a 'Music'
79 takeNotes :: Integer -> Music a -> Music a
80 takeNotes _ m@(Prim _) = m
81 takeNotes n (Modify c m) = Modify c $ takeNotes n m
82 takeNotes _ m@(_ :=: _) = m
83 takeNotes n (m1 :+: m2)
84   | n <  1    = emptyMusic
85   | n == 1    = m1
86   | otherwise = m1 :+: takeNotes (n - 1) m2
87
88 instance FromJSON Node where
89   parseJSON = withObject "Node" $ \v ->
90     Node <$> v .: "id" <*> (Prim <$> v .: "music")
91
92 lookupNode :: Text -> [Object] -> Parser Node
93 lookupNode id nodes = do
```

```
94    matches <- filterM (fmap (== id) . (.: "id")) nodes
95    case matches of
96      [node] -> parseJSON (Object node)
97      _ -> fail "Couldn't match node by id"
98
99  buildMap :: [Object] -> [Object] -> Graph -> Parser Graph
100 buildMap _ [] m = pure m
101 buildMap nodes (e:es) m = do
102   toId <- e .: "to"
103   fromId <- e .: "from"
104   edge <- Edge <$> lookupNode toId nodes <*> e .: "prob"
105   from <- lookupNode fromId nodes
106   buildMap nodes es $ insertEdge from edge m
107
108 instance FromJSON Graph where
109   parseJSON = withObject "Graph" $ \v -> do
110     edges <- v .: "edges"
111     nodes <- v .: "nodes"
112     buildMap nodes edges $ Graph mempty
113
114 instance FromJSON (Primitive Pitch) where
115   parseJSON = withObject "Primitive" $ \v -> do
116     -- TODO Ratio _Integer_ is easy DOSable
117     -- RAM consumption
118     duration <- v .: "dur"
119     octave <- v .: "octave"
120     pitchClass <- v .: "pitch"
121     case pitchClass of
122       "Rest" -> pure $ Rest duration
123       p -> pure $ Note duration (read pitchClass, octave)
```

Backend

backend/Api.hs

```haskell
{-# LANGUAGE OverloadedStrings #-}
{-# LANGUAGE FlexibleInstances #-}
{-# LANGUAGE DataKinds         #-}
{-# LANGUAGE TypeOperators     #-}
module Api where

import Data.Aeson
import Data.ByteString.Lazy (ByteString ())
import Data.Monoid ((<>))
import Data.Ratio
import Data.Text (Text ())
import GHC.Generics
import Servant.API
import Sound.Likely

type LikelyApi = "interpretation" :> Capture "format" OutputFormat
                                  :> ReqBody '[JSON] GraphWithParams
                                  :> Post '[OctetStream] ByteString
                 :<|> "seed" :> Get '[JSON] Int
                 :<|> Raw

data OutputFormat = Midi | Wav
  deriving (Show, Eq, Ord)

instance FromHttpApiData OutputFormat where
  parseUrlPiece "mid" = Right Midi
  parseUrlPiece "wav" = Right Wav
  parseUrlPiece x     = Left $ "Couldn't match " <> x <> " with {mid, wav}"

data GraphWithParams
  = GraphWithParams
  { gpParams :: Params
  , gpGraph  :: Graph
  } deriving (Show, Eq, Ord)

instance FromJSON GraphWithParams where
  parseJSON = withObject "GraphWithParams" $ \v ->
    GraphWithParams <$> v .: "params"
                    <*> v .: "graph"

data Params
  = Params
  { pMaxHops      :: Int
  , pStartingNode :: Node
  , pSeed         :: Int
  } deriving (Show, Eq, Ord)

```

```
48 instance FromJSON Params where
49   parseJSON = withObject "Params" $ \v ->
50     Params <$> v .: "maxhops"
51            <*> v .: "starting_node"
52            <*> v .: "seed"
```

backend/Main.hs

```
1  {-# LANGUAGE OverloadedStrings #-}
2  module Main where
3
4  import Api
5
6  import Codec.Midi (buildMidi)
7  import Codec.ByteString.Builder
8  import Control.Monad.IO.Class
9  import Data.ByteString.Lazy (ByteString ())
10 import qualified Data.ByteString.Lazy as B
11 import Euterpea hiding (app)
12 import GHC.IO.Handle
13 import Network.Wai
14 import Network.Wai.Handler.Warp
15 import Servant
16 import Sound.Likely
17 import System.Directory
18 import System.Exit
19 import System.Environment
20 import System.FilePath.Posix
21 import System.IO
22 import System.Process
23 import System.Random
24
25 api :: Proxy LikelyApi
26 api = Proxy
27
28 midiString :: ToMusic1 a => Music a -> ByteString
29 midiString = toLazyByteString . buildMidi . toMidi . perform
30
31 server :: Server LikelyApi
32 server = genInterpretation :<|> randomSeed :<|> serveDirectoryWebApp "web/
       dist"
33
34 randomSeed :: Handler Int
35 randomSeed = liftIO newStdGen >>= return . fst . random
36
37 genInterpretation :: OutputFormat -> GraphWithParams -> Handler ByteString
38 genInterpretation Midi g = do
39   let params       = gpParams g
40       maxHops      = fromIntegral . pMaxHops $ params
```

```
41        randomGen     = mkStdGen $ pSeed params
42        song          = interpretation randomGen (gpGraph g) (pStartingNode
              params)
43    return . midiString $ takeNotes maxHops song
44  genInterpretation Wav g = genInterpretation Midi g >>= synthWav
45
46  synthWav :: ByteString -> Handler ByteString
47  synthWav midi = do
48    inName <- tempFile "mid"
49    liftIO $ B.writeFile inName midi
50    outName <- tempFile "wav"
51    (_, _, _, ph) <- liftIO $
52      createProcess_ "fluidsynth"
53        (proc "fluidsynth"
54          [ "-a", "file"
55          , "-F", outName
56          , "-i"
57  --          , "/usr/share/soundfonts/FluidR3_GM.sf2"
58          , "/nix/store/591834mz365ccwyj3ah2d66ncsqvp8w9-Fluid-3/share/
                soundfonts/FluidR3_GM2-2.sf2"
59          , inName ])
60          { std_in = CreatePipe }
61    code <- liftIO $ waitForProcess ph
62    case code of
63      ExitFailure _ -> throwError err500 { errBody = "fluidsynth failed" }
64      ExitSuccess -> do
65        out <- liftIO $ B.readFile outName
66        liftIO $ removePathForcibly outName
67        return out
68
69  tempFile :: String -> Handler FilePath
70  tempFile ext = try 0
71    where maxtries = 100
72          try :: Integer -> Handler FilePath
73          try n
74            | n < maxtries = do
75              progName <- liftIO $ getProgName
76              let path = "/tmp" </> addExtension (makeValid progName ++ "-"
                      ++ show n) ext
77              exists <- liftIO $ doesFileExist path
78              if exists
79                then try (n + 1)
80                else pure path
81            | otherwise = throwError err500
82  app :: Application
83  app = serve api server
84
85  main :: IO ()
86  main = newStdGen >> run 8081 app
```

Web

web/source/index.html

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <meta http-equiv="x-ua-compatible" content="ie=edge" />
        <meta name="viewport" content="width=device-width, initial-scale=1" />
        <title>likely music</title>
        <link rel="stylesheet" type="text/css" href="custom.css">
        <link rel="stylesheet" type="text/css" href="vis.min.css">
        <script src="main.js"></script>
    </head>
    <body>
        <div id="network"></div>
        <div id="sidebar">
            <h1>likely music</h1>
            <h2>General Settings</h2>
            <button id="set-starting-node">Set starting node</button>
            <button id="show-starting-node">Show starting node</button>
            <h2>Generate an interpretation</h2>
            <div class="multi-inputs">
                <label for="seed">Seed:</label>
                <input type="number" id="seed">
                <button id="random-seed">&amp;#8634;</button>
            </div>
            <div class="multi-inputs">
                <label for="hop-count">Length:</label>
                <input type="number" min="0" id="hop-count" placeholder="Max. note count">
            </div>
            <div id="player-container">
                <button id="reload-player">&amp;#8634;</button>
                <audio id="player" controls></audio>
            </div>
            <div class="multi-inputs">
                <button id="download-audio">Download</button>
                <label for="format">
                    as
                </label>
                <select id="format">
                    <option value="mid">MIDI</option>
                    <option value="wav">WAV</option>
                </select>
            </div>
            <h2>Load or Save Work</h2>
            <button id="gen-score" class="save">Save</button>
            <label for="upload-score" class="custom-file">
```

```
46                  <input type="file" id="upload-score" >
47                  <span>Load</span>
48              </label>
49              <button id="clear-score" class="cancel">Clear</button>
50              <h2>Help and Links</h2>
51              <a href="https://github.com/sternenseemann/likely-music">Source
                    Code</a>
52          </div>
53          <div id="edge-overlay" class="hidden dialog">
54              <h2><span id="edge-operation"></span> edge</h2>
55              <div class="multi-inputs">
56                  <label for="prob">Probability:</label>
57                  <input id="prob" type="number" min="0.0" max="100">
58                  <span>%</span>
59              </div>
60              <div class="multi-inputs">
61                  <button class="save" id="edge-save">Save</button>
62                  <button class="cancel" id="edge-cancel">Cancel</button>
63              </div>
64          </div>
65          <div id="node-overlay" class="hidden dialog">
66              <h2><span id="node-operation"></span> node</h2>
67              <div class="multi-inputs">
68                  <label for="pitch">Pitch:</label>
69                  <select id="pitch"></select>
70              </div>
71              <div class="multi-inputs">
72                  <label for="octave">Octave:</label>
73                  <input id="octave" type="number" step="1">
74              </div>
75              <div class="multi-inputs">
76                  <label>Duration:</label>
77                  <input min="0" id="numerator" type="number" step="1">
78                  <span>/</span>
79                  <input min="0" id="denominator" type="number" step="1">
80              </div>
81              <div class="multi-inputs">
82                  <button class="save" id="node-save">Save</button>
83                  <button class="cancel" id="node-cancel">Cancel</button>
84              </div>
85          </div>
86      </body>
87  </html>
```

web/source/custom.css

```
body {
    font-size:1em;
    font-family: sans-serif;
    margin: 0px;
    background-color: black;
}

#network {
    width: 79%;
    float:left;
    height: 100vh;
}

#sidebar {
    width: 20%;
    float:right;
    color: white;
    background-color: black;
    box-shadow: 0px 0px 20px #111;
    font-size: 1.2rem;
}

#sidebar > * {
    width: 100%;
    border-top: 1px solid #232200;
    color: white;
    padding-left: 0px;
    padding-right: 0px;
    margin: 0;
}

#sidebar button:hover, #sidebar input:hover,
#sidebar .custom-file:hover, #sidebar select:hover, #sidebar a:hover {
    background-color: #563d7c;
}

#sidebar button, #sidebar input, #sidebar .custom-file, #sidebar select, #
    sidebar a {
  background-color: #000;
}

#sidebar h1 {
    font-size: 1.5rem;
    padding-top: 0.75rem;
    padding-bottom: 0.75rem;
    text-align: center;
    background-color: #111;
}
```

```
48
49 #sidebar h2 {
50     font-size: 1.2rem;
51     padding-top: 0.9rem;
52     padding-bottom: 0.9rem;
53     text-align: center;
54     background-color: #222;
55 }
56
57 #sidebar select {
58   color: white;
59   border: none;
60   padding: 0.75rem;
61   font-size: 1.2rem;
62   width: auto;
63 }
64
65 #sidebar a {
66   padding: 0.75rem;
67   display: inline-block;
68   text-decoration: none;
69   color: white;
70   text-align: center;
71 }
72
73 button {
74     border: none;
75     color: white;
76     background-color:black;
77     font-size: 1.2rem;
78     margin:0;
79     padding:0.75rem;
80 }
81
82 input[type="number"] {
83     background-color: #333;
84     color: white;
85     border: none;
86     text-align: center;
87     font-size: 1.2rem;
88     padding:0.75rem;
89 }
90
91 .custom-file {
92     top:0;
93     right:0;
94     position: relative;
95     display: inline-block;
96     height: 3rem;
97 }
```

```
98
99  .custom-file input[type="file"] {
100     position: relative;
101     top:0;
102     left:0;
103     right:0;
104     z-index:0;
105     opacity: 0;
106     width: 100%;
107     height: 100% !important;
108     margin:0;
109     padding:0;
110 }
111
112 .custom-file span {
113     text-align: center;
114     position: absolute;
115     top: 0;
116     left: 0;
117     right: 0;
118     z-index: 1;
119     width: 100%;
120     height: 3rem;
121     pointer-events: none;
122     background-color: transparent !important;
123     font-size: 1.2rem;
124     line-height: 1.5rem;
125     padding-top: 0.75rem;
126     padding-bottom: 0.75rem;
127 }
128
129 .dialog {
130     position: absolute;
131     top: 10%;
132     left: 25%;
133     width: 30%;
134     min-width:500px;
135     padding: 10px;
136     background-color: black;
137     color: white;
138     box-shadow: 0px 0px 10px #111;
139 }
140
141 .dialog > div {
142     height: 3rem;
143 }
144
145 .hidden {
146     visibility:hidden;
147 }
```

```
148
149 .dialog > div {
150   width: 100%;
151 }
152
153 .dialog button {
154     padding: 0.75rem;
155     font-size: 1.5rem;
156 }
157
158 button.cancel {
159     background-color: #a23a30;
160 }
161
162 button.save {
163     background-color: #0ea92f;
164 }
165
166 .dialog .multi-inputs {
167   font-size: 1.5rem;
168 }
169
170 .multi-inputs {
171   display: inline-flex;
172   flex-direction: row;
173   flex-wrap: nowrap;
174   justify-content: flex-start;
175   align-items: baseline;
176   width: 100%;
177 }
178
179 .multi-inputs > * {
180   flex-grow: 1;
181   flex-basis: auto;
182   transition: width 0.7s ease-out;
183   max-height: 100%;
184   text-align: center;
185 }
186
187 .multi-inputs :nth-child(1) {
188   text-align: left;
189 }
190
191 .multi-inputs label {
192   display: inline-block;
193   background-color: #333;
194   padding: 0.75rem;
195 }
196
197 .multi-inputs input {
```

```
198     display: inline-block;
199     color: white;
200     background-color: #111;
201     padding: 0.75rem;
202     border: none;
203     min-width: 0px;
204   }
205
206   .multi-inputs span {
207     display: inline-block;
208     padding: 0.75rem;
209     background-color: #222;
210   }
211
212   .multi-inputs button {
213       padding: 0.75rem;
214   }
215
216   #player-container {
217     display: inline-flex;
218     align-items: center;
219   }
220
221   #player-container > * {
222     flex: auto;
223   }
```

web/source/main.js

```
1  import vis from 'vis';
2  import { Map } from 'immutable';
3  // types / internals
4
5  const valid_pitches = [
6      'Rest',
7      'Cff', 'Cf', 'C',
8      'Dff', 'Cs', 'Df',
9      'Css', 'D', 'Eff',
10     'Ds', 'Ef', 'Fff',
11     'Dss', 'E', 'Ff',
12     'Es', 'F', 'Gff',
13     'Ess', 'Fs', 'Gf',
14     'Fss', 'G', 'Aff',
15     'Gs', 'Af', 'Gss',
16     'A', 'Bff', 'As',
17     'Bf', 'Ass', 'B',
18     'Bs', 'Bss'
19 ];
20
21 const display_pitches = [
22     'Rest',
23     ' C', 'C', 'C',
24     ' D', 'C', 'D',
25     'C', 'D', ' E',
26     'D', 'E', ' F',
27     'D', 'E', 'F',
28     'E', 'F', 'Gff',
29     'E', 'F', 'G',
30     'F', 'G', ' A',
31     'G', 'A', 'G',
32     'A', ' B', 'A',
33     'B', 'A', 'B',
34     'B', 'B'
35 ];
36
37 function displayPitch(pitch) {
38     var i = valid_pitches.indexOf(pitch);
39     if(i === -1) {
40         throw 'Invalid pitch';
41     } else {
42         return display_pitches[i];
43     }
44 }
45
46 function standard_rests(dur) {
47     if(dur.numerator === 1) {
48         switch(dur.denominator) {
```

```
49              case 1:
50                  return  '';
51                  break;
52              case 2:
53                  return  '';
54                  break;
55              case 4:
56                  return  '';
57                  break;
58              case 8:
59                  return  '';
60                  break;
61              case 16:
62                  return  '';
63                  break;
64              case 32:
65                  return  '';
66                  break;
67              case 64:
68                  return  ''
69                  break;
70              case 128:
71                  return  ''
72                  break;
73              default:
74                  return null;
75                  break;
76          }
77      } else {
78          return null;
79      }
80  }
81
82  function standard_notes(dur) {
83      if(dur.numerator === 1) {
84          switch(dur.denominator) {
85              case 1:
86                  return  '';
87                  break;
88              case 2:
89                  return  '';
90                  break;
91              case 4:
92                  return  '';
93                  break;
94              case 8:
95                  return  '';
96                  break;
97              case 16:
98                  return  '';
```

```
99                  break;
100             case 32:
101                 return  '';
102                 break;
103             case 64:
104                 return  ''
105                 break;
106             case 128:
107                 return  ''
108                 break;
109             default:
110                 return null;
111                 break;
112         }
113     } else if(dur.numerator === 2 && dur.denominator === 1) {
114         return  ''
115     } else {
116         return null;
117     }
118 }
119
120 function compute_dot_times(dur, denominator) {
121     let baseLog = (b, x) => Math.log(x) / Math.log(b);
122     let term = (dur.numerator * Math.pow(2, denominator)) / dur.denominator
        ;
123     return [ denominator, baseLog(1.5, term)];
124 }
125
126 function musical_symbol(lookup, dur) {
127     const dot =  ' ';
128     let isNat = n => {
129         if (typeof n !== 'number')
130             return false;
131         return (n >= 0.0) && (Math.floor(n) === n) && n !== Infinity;
132     };
133     var standard_symbol = lookup(dur);
134     var bla = [0, 1, 2, 3, 4, 5, 6, 7 ].map(compute_dot_times.bind(dur));
135     console.log(bla);
136     var dots = bla.filter(([den, dots]) => isNat(dots));
137
138     console.log(dots);
139
140     if(standard_symbol !== null) {
141         return standard_symbol;
142     } else if (dots.length !== 0) {
143         var symbol = lookup(new Rational(1, dots[0][0]));
144         for(var i = dots[0]; i > 0; i--) {
145             symbol = symbol + dot;
146         }
147         return symbol;
```

```
148     } else {
149         return dur.toString();
150     }
151 }
152
153 class Music {
154     constructor(dur, pitch_class, octave) {
155         this.dur = dur;
156         if(valid_pitches.indexOf(pitch_class) !== -1) {
157             this.pitch = pitch_class;
158         } else {
159             throw `Invalid pitch class '${pitch_class}'`;
160         }
161         this.octave = octave;
162     }
163
164     toString() {
165         if(this.pitch === 'Rest') {
166             return `${displayPitch(this.pitch)} for ${this.dur.toString()
                    }`;
167         } else {
168             return `${displayPitch(this.pitch)}${this.octave} for ${this.
                    dur.toString()}`;
169         }
170     }
171
172     nodeText() {
173         if(this.pitch === 'Rest') {
174             // alignment using a space! #justvisjsthings
175             return ` ${musical_symbol(standard_rests, this.dur)}`;
176         } else {
177             return `${musical_symbol(standard_notes, this.dur)}   ${
                    displayPitch(this.pitch)}${this.octave}`
178         }
179     }
180
181
182     static fromObject(obj) {
183         return new Music(Rational.fromObject(obj.dur), obj.pitch, Number(
                obj.octave));
184     }
185 }
186
187 class Rational {
188     constructor(a, b) {
189         this.numerator = a;
190         this.denominator = b;
191         this.reduce();
192     }
193
```

```
194     reduce() {
195         let gcd = (a, b) => !b ? a : gcd(b, a % b);
196         let div = function(a, b) {
197             if(b === 0) {
198                 throw 'Divide by zero';
199             } else {
200                 return Math.floor(a / b);
201             }
202         };
203 
204         var d = gcd(this.numerator, this.denominator);
205         this.numerator = div(this.numerator, d);
206         this.denominator = div(this.denominator, d);
207     }
208 
209     toString() {
210         return `${this.numerator}/${this.denominator}`;
211     }
212 
213     static fromObject(obj) {
214         return new Rational(obj.numerator, obj.denominator);
215     }
216 }
217 
218 function collectGraphData(nodeDate, edgeData) {
219     return {
220         nodes: [... nodeData.values()].map(x => ({
221             id: x.nodeData.id,
222             music: x.music
223         })),
224         edges: [... edgeData.values()].map(x => ({
225             id: x.edgeData.id,
226             from: x.edgeData.from,
227             to: x.edgeData.to,
228             prob: x.prob
229         }))
230     };
231 }
232 
233 function importGraphData(g) {
234     nodeData = new Map();
235     edgeData = new Map();
236     var nodeSet = new vis.DataSet({});
237     var edgeSet = new vis.DataSet({});
238     for(let node of g.nodes) {
239         var music = Music.fromObject(node.music);
240         var data = { id: node.id, label: music.nodeText() };
241         nodeData = nodeData.set(node.id, { nodeData: data, music: node.
                music });
242         nodeSet.add(data);
```

```
243     }
244
245     for(let edge of g.edges) {
246         var data = {
247             id: edge.id,
248             from: edge.from,
249             to: edge.to,
250             label: `${edge.prob * 100}%`
251         };
252         edgeData = edgeData.set(edge.id, { edgeData: data, prob: edge.prob
                });
253         edgeSet.add(data);
254     }
255
256     network.setData({ nodes: nodeSet, edges: edgeSet });
257 }
258
259 // helper
260
261 function download(url, filename) {
262     var link = document.createElement('a');
263     link.setAttribute('href', url);
264     link.setAttribute('download', filename);
265     link.style.display = 'none';
266     document.body.appendChild(link);
267     link.click();
268     document.body.removeChild(link);
269 }
270
271 function downloadFile(content_type, filename, content) {
272     var data = `data:${content_type},${encodeURIComponent(content)}`;
273     download(data, filename);
274 }
275
276
277 // graph code
278
279 var nodeData = Map();
280 var edgeData = Map();
281 var network = null;
282 var starting_node_id = null;
283
284
285 function showOverlay(id) {
286     document.getElementById(id).classList.remove('hidden');
287 }
288
289 function genericEditNode(data, callback) {
290     function clearOverlay() {
291         document.getElementById('node-save').onclick = null;
```

```
292         document.getElementById('node-cancel').onclick = null;
293         hideOverlay('node-overlay');
294     }
295
296     function saveNode(data, callback) {
297         var duration = new Rational(document.getElementById('numerator').
                value,
298             document.getElementById('denominator').value);
299         var music = new Music(duration, document.getElementById('pitch').
                value,
300             Number(document.getElementById('octave').value));
301         data.label = music.nodeText();
302         clearOverlay();
303         callback(data);
304         nodeData = nodeData.set(data.id, { music: music, nodeData: data });
305     }
306
307     function discardNode(callback) {
308         clearOverlay();
309         callback(null);
310     }
311
312     showOverlay('node-overlay');
313     var node = nodeData.get(data.id);
314     if(node !== undefined) {
315         var music = node.music;
316         document.getElementById('pitch').value = music.pitch;
317         document.getElementById('octave').value = music.octave;
318         document.getElementById('numerator').value = music.dur.numerator;
319         document.getElementById('denominator').value = music.dur.
                denominator;
320     }
321     document.getElementById('node-save').onclick = saveNode.bind(this, data
            , callback);
322     document.getElementById('node-cancel').onclick = discardNode.bind(this,
             callback);
323 }
324
325 function genericEditEdge(data, callback) {
326     function clearOverlay() {
327         document.getElementById('edge-save').onclick = saveEdge.bind(this,
                data, callback);
328         document.getElementById('edge-cancel').onclick = discardEdge.bind(
                this, callback);
329         hideOverlay('edge-overlay');
330     }
331
332     function saveEdge(data, callback) {
333         // for some reason, editWithoutDrag
334         // sets from & to to the node respective
```

```
335         // node objects, which results in the edge
336         // disappearing.
337         if (typeof data.to === 'object')
338             data.to = data.to.id
339         if (typeof data.from === 'object')
340             data.from = data.from.id
341 
342         var prob = document.getElementById('prob').value / 100;
343         data.label = `${prob * 100}%`;
344         clearOverlay();
345         callback(data);
346         edgeData = edgeData.set(data.id, { prob: prob, edgeData: data } );
347     }
348 
349     function discardEdge(callback) {
350         clearOverlay();
351         callback(null);
352     }
353 
354     showOverlay('edge-overlay');
355     var edge = edgeData.get(data.id);
356     if(edge !== undefined) {
357         document.getElementById('prob').value = edge.prob * 100;
358     }
359     document.getElementById('edge-save').onclick = saveEdge.bind(this, data
        , callback);
360     document.getElementById('edge-cancel').onclick = discardEdge.bind(this,
         callback);
361 }
362 
363 function deleteFromMap(data, callback) {
364     for(let node of data.nodes) {
365         nodeData = nodeData.delete(node);
366     }
367 
368     for(let edge of data.edges) {
369         edgeData = edgeData.delete(edge);
370     }
371 
372     callback(data);
373 }
374 
375 
376 function hideOverlay(id) {
377     document.getElementById(id).classList.add('hidden');
378 }
379 
380 function handleImport() {
381     var files = document.getElementById('upload-score').files;
382     if(files.length === 0) {
```

```
383         alert('Select a file first!');
384     } else {
385         var file = files[0];
386         var reader = new FileReader();
387         reader.addEventListener('loadend', function() {
388             var parsed = JSON.parse(this.result);
389             if(parsed === undefined) {
390                 alert('Could not parse likely score');
391             } else {
392                 var confirmation = window.confirm('Proceeding will
                        overwrite the current graph. Are you sure?');
393                 if(confirmation) {
394                     try {
395                         importGraphData(parsed);
396                     } catch(e) {
397                         alert(`Could not import likely score, probably the
                            file was malformed. Error: ${e}`);
398                     }
399                 }
400             }
401         });
402         reader.readAsText(file);
403     }
404 }
405
406 function saveDataToLocalStorage() {
407     const json = JSON.stringify(collectGraphData(nodeData, edgeData));
408     const params = JSON.stringify(gatherParams());
409     localStorage.setItem("score", json)
410     localStorage.setItem("params", params)
411 }
412
413 function showStartingNode() {
414     if(typeof starting_node_id === 'string') {
415         network.selectNodes([starting_node_id], false);
416     } else {
417         alert('No starting node selected yet!');
418     }
419 }
420
421 function setStartingNode() {
422     var selected = network.getSelectedNodes();
423     if(selected.length > 1) {
424         alert('Only select one node!');
425     } else if(selected.length === 0) {
426         alert('Select a node first!');
427     } else {
428         starting_node_id = selected[0];
429     }
430 }
```

```
431
432 function fetchInterpretation(params, format) {
433     var jsonRequest = JSON.stringify({
434         graph: collectGraphData(nodeData, edgeData),
435         params: params
436     });
437
438     var myHeaders = new Headers();
439     myHeaders.set('Content-Type', 'application/json');
440
441     var myInit = {
442         method: 'POST',
443         headers: myHeaders,
444         mode: 'cors',
445         body: jsonRequest
446     };
447
448     var myRequest = new Request(`/interpretation/${format}`, myInit);
449
450     return fetch(myRequest).then(res => res.blob());
451 }
452
453 function gatherParams() {
454     var starting_node_entry = nodeData.get(starting_node_id);
455     if(starting_node_entry !== undefined && starting_node_entry !== null) {
456         var starting_node = {
457             id: starting_node_entry.nodeData.id,
458             music: starting_node_entry.music
459         };
460     } else {
461         var starting_node = null
462     }
463
464     var maxhops = document.getElementById('hop-count').value;
465     if(maxhops === "" || Number(maxhops) === NaN) {
466         maxhops = null;
467     } else {
468         maxhops = Number(maxhops);
469     }
470
471     var seed = document.getElementById('seed').value;
472     if(seed === "" || Number(seed) === NaN) {
473         seed = null;
474     } else {
475         seed = Number(seed);
476     }
477
478     return {
479         maxhops:  maxhops,
480         starting_node: starting_node,
```

```
481             seed: seed
482         };
483     }
484
485     function completeGatherParams() {
486         var p = gatherParams();
487         if(p.starting_node === null) {
488             alert('Set a starting node first!');
489             return null;
490         }
491
492         if(p.maxhops === null) {
493             alert('Set the maximum amount of hops to a valid number');
494             return null;
495         }
496
497         if(p.seed === null) {
498             // TODO auto generate a random one, let the user confirm before
499             alert('Set the seed to a valid number!');
500             return null;
501         }
502
503         return p;
504     }
505
506     function importParams(p) {
507         if(p.starting_node !== null) {
508             starting_node_id = p.starting_node.id;
509         }
510         if(p.seed !== null) {
511             document.getElementById('seed').value = p.seed;
512         }
513         if(p.maxhops !== null) {
514             document.getElementById('hop-count').value = p.maxhops;
515         }
516     }
517
518     function randomSeed() {
519         if(window.crypto) {
520             var array = new Int32Array(1);
521             window.crypto.getRandomValues(array);
522             document.getElementById('seed').value = array[0];
523         }
524     }
525
526     function downloadInterpretation(format) {
527         var params = completeGatherParams();
528         if(params != null) {
529             try {
530                 fetchInterpretation(params, format).then(file => {
```

```
531                 var url = URL.createObjectURL(file);
532                 download(url, `export.${format}`);
533             });
534         } catch(e) {
535             alert('An error occured while contacting the API: ' + e);
536         }
537     }
538 }
539
540 function reloadPlayer() {
541     var params = completeGatherParams();
542     if(params !== null) {
543         document.getElementById('player').src = null;
544         try {
545             fetchInterpretation(params, 'wav').then(file => {
546                 var url = URL.createObjectURL(file);
547                 document.getElementById('player').src = url;
548             });
549         } catch(e) {
550             alert('An error occured while contacting the API: ' + e);
551         }
552     }
553 }
554
555 function init() {
556     var container = document.getElementById('network');
557
558     var options = {
559         manipulation: {
560             addNode: function(nodeData, callback) {
561                 document.getElementById('node-operation').innerHTML = 'Add
                        ';
562                 genericEditNode(nodeData, callback);
563             },
564             addEdge: function(edgeData, callback) {
565                 document.getElementById('edge-operation').innerHTML = 'Add
                        ';
566                 genericEditEdge(edgeData, callback);
567             },
568             editNode: function(nodeData, callback) {
569                 document.getElementById('node-operation').innerHTML = 'Edit
                        ';
570                 genericEditNode(nodeData, callback);
571             },
572             editEdge: {
573                 editWithoutDrag: function(edgeData, callback) {
574                     document.getElementById('edge-operation').innerHTML = '
                            Edit';
575                     genericEditEdge(edgeData, callback);
576                 }
```

```
577             },
578             deleteNode: deleteFromMap,
579             deleteEdge: deleteFromMap,
580             controlNodeStyle: {
581             }
582         },
583         nodes: {
584             borderWidth: 0,
585             color: {
586                 background: '#563d7c',
587                 hover: {
588                     background: '#8f14ff'
589                 },
590                 highlight: {
591                     background: '#8f14ff'
592                 }
593             },
594             chosen: true,
595             font: {
596                 color: 'white',
597                 size: 20,
598                 align: 'center'
599             },
600             shape: 'circle',
601         },
602         edges: {
603             arrows: {
604                 to: { enabled: true }
605             },
606             color: {
607                 color: '#563d7c',
608                 hover: '#563d7c',
609                 highlight: '#563d7c',
610             },
611             font: {
612                 color: '#ffffff',
613                 strokeWidth: 0
614             }
615         }
616     };
617
618     network = new vis.Network(container, {}, options);
619
620     try {
621         const score = localStorage.getItem('score');
622         if(score !== null) {
623             importGraphData(JSON.parse(score));
624         }
625     } catch(e) {
626         localStorage.removeItem('score');
```

```
627     }
628
629     try {
630         const params = localStorage.getItem('params')
631         if(params !== null) {
632             importParams(JSON.parse(params));
633         }
634     } catch(e) {
635         localStorage.removeItem('params');
636     }
637
638     const pitch_selector = valid_pitches.map((p, i) =>
639         `<option value="${p}">${display_pitches[i]}</option>`)
640         .reduce((acc, v) =>
641             acc + v, '');
642     document.getElementById('pitch').innerHTML = pitch_selector;
643
644     /* event handling, order as in sidebar */
645     document.getElementById('set-starting-node').onclick = setStartingNode;
646     document.getElementById('show-starting-node').onclick =
        showStartingNode;
647
648     document.getElementById('random-seed').onclick = randomSeed;
649
650     document.getElementById('reload-player').onclick = reloadPlayer;
651     document.getElementById('download-audio').onclick = () => {
652         var format = document.getElementById('format').value;
653         downloadInterpretation(format);
654     };
655
656     document.getElementById('gen-score').onclick = () =>
657         downloadFile('application/json', 'score.likely.json',
658             JSON.stringify(collectGraphData(nodeData, edgeData)));
659     document.getElementById('upload-score').addEventListener('change',
        handleImport);
660     document.getElementById('clear-score').onclick = () =>
661         importGraphData({ nodes: [], edges: []});
662
663     window.setInterval(saveDataToLocalStorage, 5000);
664 }
665
666 document.addEventListener('DOMContentLoaded', () => init());
```

Lizenz

Preamble

The GNU Affero General Public License is a free, copyleft license for software and other kinds of works, specifically designed to ensure cooperation with the community in the case of network server software.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, our General Public Licenses are intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

Developers that use our General Public Licenses protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License which gives you legal permission to copy, distribute and/or modify the software.

A secondary benefit of defending all users' freedom is that improvements made in alternate versions of the program, if they receive widespread use, become available for other developers to incorporate. Many developers of free software are heartened and encouraged by the resulting cooperation. However, in the case of software used on network servers, this result may fail to come about. The GNU General Public License permits making a modified version and letting the public access it on a server without ever releasing its source code to the public.

The GNU Affero General Public License is designed specifically to ensure that, in such cases, the modified source code becomes available to the community. It requires the operator of a network server to provide the source code of the modified version running there to the users of that server. Therefore, public use of a modified version, on a publicly accessible server, gives the public access to the source code of the modified version.

An older license, called the Affero General Public License and published by Affero, was designed to accomplish similar goals. This is a different license, not a version of the Affero GPL, but Affero has released a new version of the Affero GPL which permits relicensing under this license.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

"This License" refers to version 3 of the GNU Affero General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

   A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

   The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

   The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

   All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

   You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for

you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

   No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

   When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

   (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

   (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

   (d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

   A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

(a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

(b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

(c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

(d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

(e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

(a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

(b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

(c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

(d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

(e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

(f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

   You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

    If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Remote Network Interaction; Use with the GNU General Public License.

    Notwithstanding any other provision of this License, if you modify the Program, your modified version must prominently offer all users interacting with it remotely through a computer network (if your version supports such interaction) an opportunity to receive the Corresponding Source of your version by providing access to the Corresponding Source from a network server at no charge, through some standard or customary means of facilitating copying of software. This Corresponding Source shall include the Corresponding Source for any work covered by version 3 of the GNU General Public License that is incorporated pursuant to the following paragraph.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the work with which it is combined will remain governed by version 3 of the GNU General Public License.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU Affero General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU Affero General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU Affero General Public License, you may choose any version ever published by the Free Software Foundation.

    If the Program specifies that a proxy can decide which future versions of the GNU Affero General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

    Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A

PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

End of Terms and Conditions

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <textyear>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU Affero General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If your software can interact with users remotely through a computer network, you should also make sure that it provides a way for users to get its source. For example, if your program is a

web application, its interface could display a "Source" link that leads users to an archive of the code. There are many ways you could offer source, and different solutions will be better for different programs; see section 13 for the specific requirements.

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU AGPL, see `http://www.gnu.org/licenses/`.