Äänet ja musiikki Haskellilla

Joni Purojärvi
jopuroja@jyu.fi

28.4.2008

Tiivistelmä

Ääntä ja musiikkia voidaan tuottaa joko suoraan soittamalla valmiita tiedostoja tai synteettisesti äänikortin avulla (esimerkiksi MIDI). Tässä paperissa tutustutaan äänen tuottamiseen yleisesti ja esitellään Haskellin äänikirjastoja. Erityisesti perehdytään Haskore-kirjastoon, jonka avulla voidaan suhteellisen yksinkertaisesti kuvata länsimaalaista musiikkia.

1 Johdanto

Ääntä ja musiikkia ei suoraan saa tuotettua Haskellin peruskirjastoilla. Tätä varten on kuitenkin tehty jonkin verran kirjastoja niin äänen tuottamiseen, signaalin käsittelyyn ja jopa musiikin kuvaamiseen. Tässä raportissa esitellään lyhyesti yleiset periaatteet äänen tuottamisesta (Haskellin) kannalta, muutama Haskell-kirjasto yleisesti sekä lopuksi tarkemmin Haskore- ja siihen pohjautuva HasChorus-kirjastot.

2 Yleistä äänen tuottamisesta

Ääntä voidaan tuottaa suoraan äänikortin kautta syntetisoimalla tai soittamalla valmiita äänitiedostoja. Joka tapauksessa äänentuottamiseen vaaditaan siihen erikoistunut äänipiiri ja tuki käyttöjärjestelmältä. Perinteisesti ääntä tuotettiin syntetisoimalla ääniä, mutta nykyisin valtavirran ohjelmistoissa ja peleissä soitetaan suoraan erilaisia valmiita äänitiedostoja. Varhaisessa vaiheessa ääntä tuotettiin yksinomaa syntetisoimalla[1].

IBM PC -yhteensopivia äänikortit alkoivat yleistymään vasta 1980-luvun lopussa ja 1990-luvun alussa. Ennen äänikorttien valtakautta ainoa tapa saada PC:stä ääntä ulos oli ”PC speaker” eli ns. piipperi. PC:n piipperi oli varsin

alkeellinen. Yksi mainitsemisen arvoinen äänipiiri on SID, josta ehkä suurimmalle yleisölle tutuin on Commodore 64:n mukana tullut MOS 6581 -piiri.

Ääntä saadaan ohjelmistoihin ja peleihin erilaisten äänikirjastojen kautta. Haskelille on muutamia tällaisia kirjastoja, joilla voidaan tuottaa suoraan ääntä äänikortin avulla. Tällainen on esimerkiksi yampasynth-äänikirjasto joka käyttä OpenAL-kirjastoa rajapintanaan äänikortille [3].

Suoraan syntentisoimalla tiettyjä ääniä eri korkeuksina ei ole aina sitä mitä halutaan. Tämän vuoksi olisi hyvä voida kuvata ääntä jollakin tapaa. Ääni luodaan vielä jotakuin syntetisoimalla, mutta tarkoituksensa on saada jotakin oikeaa soitinta muistuttavaa ääntä. Tällainen on esimerkiksi MIDI.

Haskoressa musiikki kuvataan aluksi Haskoren esitystavalla, jota Paul Hudak kutsuu Haskell School of Expression -kirjassa MDL:ksi, Music description language [4]. Tämä tulkataan edelleen MIDIn ymmärtämäksi muodoksi.

Haskelille on tehty myös kirjastoja, joidenka avulla voidaan soittaa ogg, wav ja mp3-tiedostoja. Näiden kirjastojen avulla on toteutuuttu muutamia stand-alone soittimia, kuten hmp3 ja HOgg.

3 Hackagen äänikirjastoista

Haskellin hackage-kirjostoista[2] löytyy muutamia äänikirjastoja, jotka antavat rajapinnan äänen käsittelyyn. Tällaisia kirjastoja ovat mm. ALUT, joka mahdollistaa OpenAL:n käytön (Open Audio Library) ja ALSA-midi (Advanced Linux Sound Architecture). Jälkimmäisen kehittäminen on tosin osittain lopetettu, koska kehitteillä on rajapinnan luonti suoraan ALSAlle. OpenAL on ainakin osin toteutettu käyttämällä valmiita C-kirjastokutsuja.

Hackagesta löytyy myös muutamalle ääniformaatille suora tuki, kuten WAV, MP3 ja OGG-tiedostoille. Näiden avulla on toteutettu muutamia valmiita stand-alone -soittimia, kuten edellisessä kappaleessa mainittiin.

Tarjolla on myös yksi kokeellinen ohjelmistosyntetisaattori, YampaSynth. Tämän avulla on mahdollista tuottaa ääntä ja musiikkia joko suoraan äänikortille reaaliajassa tai tallentaa tuotos WAV-tiedostoon.

Tässä paperissa keskitytään kuitenkin Haskoreen ja sen päälle kehitettyyn HasChorukseen.

4 Haskore

Haskore on Paul Hudakin ja muutamien avustajien toteuttama laaja kokoelma Haskell-moduuleita, joilla voidaan helpohkosti kuvata musiikin rakenteita

korkealla tasolla. Sellaisenaan se sopii hyvin länsimaalaisen musiikin kuvaamiseen.

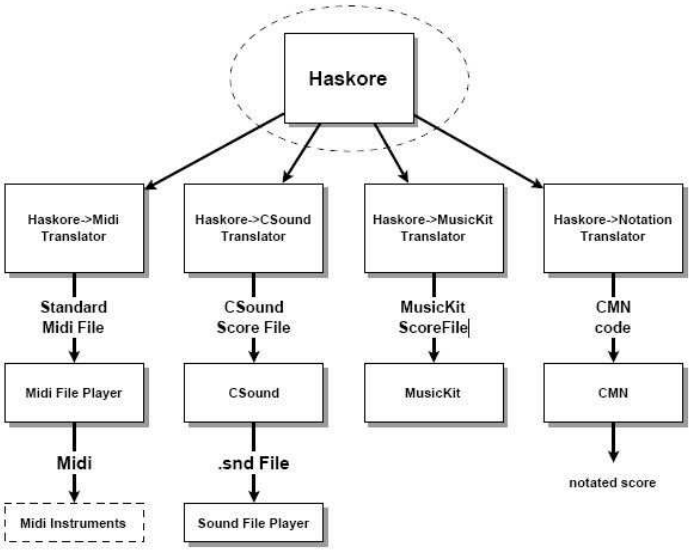
Alunperin Haskoren kehitys loppui vuonna 2000, mutta sittemmin sitä on kehitetty laajalti vapaaehtoisvoimin. Uusin versio ei ole enää ole rakenteeltaan aivan samanlainen kuin versio, joka tässä paperissa esitetään. Vanha versio ei enää toimi kunnolla uusimmalla GHC:llä Windowsilla johtuen File.IO:n muuttumisesta.

Tätä paperia varten käytettiin Hugsin versiota 20050308 Linuxilla ja Haskoren syyskuussa 2000 julkaistua versiota.

Haskorella kuvataan musiikin rakenne soinnuin ja perinteisin länsimaalaisin tavoin, jotka ajetaan Haskoren esitystavasta MIDI-tiedostoksi. Alunperin Haskoreen oli tarkoitus saada täysi tuki myös CSound, NeXT MusicKit ja Common music notation -esitystavoille, mutta vain MIDIin saatiin toteutettua tarpeeksi hyvin. Myöhemmin CMN:n toteuttamisesta luovuttiin kokonaan.

5 Haskoren rakenne

Kuvassa 1 on esitelty Haskoren rakenne siten kuin se oli Paul Hudakin tutoriaaliin [5] kirjoittamassa versiossa. Sama perusidea on uusimmassakin Haskoren versiossa.



Kuva 1: Haskoren rakenne [5].

Haskore koostuu ytimestä, jonka ympärille ladataan erilaisia ”kääntäjiä”.

Kääntäjillä rakennetaan niiden edustamien tiedostomuotojen tyylisiä tiedostoja, joita voidaan soittaa millä tahansa niitä tukevia tiedostoja. Rakenne toteuttaa ainakin osin tulkkiarkkitehtuurin.

Haskorella esitetään musiikki rakenteineen ja nämä tiedostot voidaan kääntää halutuiksi soitettavaksi tiedostoiksi. Raportissa käytetyllä versiolla saa vain MIDI-tiedostoja, mutta Haskoren arkkitehtuuri sallii muidenkin tiedostyyppien lisäämisen suhteellisen kivuttomasti. Uusimpaan epävakaaseen versioon [7] on lisätty osittainen CSound ja SuperCollider tuki.

6 Musiikin esittäminen Haskoressa

Haskoressa esitetään säveletasot (Pitch) pareina, jotka muodostuvat korkeusluokasta ja oktaavista. Itse musiikki kuvataan kokonaisuutena, joka muodostuu nuoteista, tauoista ja temposta. Lisäksi music-datassa voidaan esittää soitettavan instrumentin nimi, jonka on löydyttävä MIDI-kirjastossa esitellyistä soittimista [8].

```
type Pitch  = (PitchClass, Octave)
data PitchClass = Cf | C | Cs | ... | Bs
type Octave = Int
```

Alla on esitelty Music-tietotyyppi, jonka avulla voidaan esittää yhden soittimen musiikki.

```
data Music = Note Pitch Dur [NoteAttribute]   -- a note \ atomic
           | Rest Dur                         -- a rest /    objects
           | Music :+: Music                  -- sequential composition
           | Music :=: Music                  -- parallel composition
           | Tempo  (Ratio Int) Music         -- scale the tempo
           | Trans  Int Music                 -- transposition
           | Instr  IName Music               -- instrument label
           | Player PName Music               -- player label
           | Phrase [PhraseAttribute] Music   -- phrase attributes
    deriving (Show, Eq)

type Dur   = Ratio Int                        -- in whole notes
type IName = String
type PName = String
```

Soinnut voidaan esittää Haskoren avulla helposti soitettuan yhtä aikaa (chord) tai murtoisointuna (arpeggio). Murtosoinnun saa aikaiseksi soitta-

malla jokainen ääni erikseen peräkkäin ja kokosoinnun saa aikaseksi soittamalla kaikki äänet yhtä aikaan. Tätä varten Haskoressa on esitelty seuraavat funktiot:

```
line, chord :: [Music] -> Music
line = foldr1 (:+:)
chord = foldr1 (:=:)
```

Line-funktiossa käytetty `(:+:)` soittaa äänet peräkkäin kun taas `(:=:)` pakottaa äänet soimaan saman aikaisesti. Näitä funktioita voidaan käyttää myös muihinkin kuin pelkkien sointujen muodostamiseen.

Sointujen voidaan kansantajuisesti sanoen muodostuvan ”samaan aikaan soitetuista sävelistä”. Muodostetaan esimerkin vuoksi kolme voimasointua ja ”soitetaan” ne peräkkäin. Voimasoinnut muodostuvat pohjasävelestä ja puhtaasta kvintistä.

```
--A5
 a5 = [ n 4 qn [] | n <- [a,e] ]
 a5chord = chord a5

--G5
 g5 = [ n 4 qn [] | n <- [g,d] ]
 g5chord = chord g5

--E5
 e5 = [ n 4 qn [] | n <- [e,b] ]
 e5chord = chord e5
--eli saatiin TNT
a5:+:g5:+:e5
```

Näiden lisäksi Haskoressa voidaan esittää tehokkaasti myös lyömäsoittimet ja muu tarpeellinen. Nämä kaikki kasataan Music-tyypiksi.

MIDIn ollessa kyseessä voidaan jokaiselle äänelle antaa oma soitin. Kaikki MIDIssä [8] määritellyt soittimet ovat käytettävissä suoraan Haskoren GeneralMidi.lhs:stä.

MIDI-tiedoston voi myös kääntää takaisin Haskoren ymmärtämään esitystapaan.

7 MIDIN tuottaminen Haskorella

Haskorella kuvatun musiikin, joka on Music-muodossa, voidaan kääntää helposti MIDI-tiedostoksi. Musiikin täytyy olla kuvattuna – yleensä omassa

tiedostossa– kokonaan kaikkine tietoineen, jotta sen kääntäminen on mahdollista.

Esimerkiksi Haskoren mukana tulee osittainen toteutus Chic Corean kappaleesta "Children's chong 6". Sen saa ladattua Hugsiin käynnistämällä komentoriviltä `Hugs childSong6.lhs`, joka lataa mukaan myös kaiken MIDI-tiedostoksi muuttamista varten.

Kappaleen esityksen voi tulostaa yksinkertaisesti kirjoittamalla `childSong6`, joka on music-tyyppiä. Music-tyypistä saa MIDI-tiedoston esimerkiksi kirjoittamalla Hugsin konsoliin `test (Instr "piano"childSong6)`, joka asettaa kappaleen instrumentiksi pianon ja tallentaa MIDI-tiedoston nimellä `test.mid`.

8 HasChorus

HasChorus Martin Schwenken toteuttama moduulikirjasto, joka pohjautuu Haskoreen [9]. Sen avulla voidaan esittää hieman Haskorea yksinkertaisesti peräkkäisesti toistuvista melodioista rakentuvaa musiikkia.

HasChoruksessa tulee mukana helppokäyttöiset EasyNote ja EasyMusic, jolla voidaan kuvata helposti yksinkertaisia sointukulkuja. Nämä voidaan muuttaa Music-tyypiksi makeNoten avulla.

```
type EasyNote  = (AbsPitch, Dur)
type EasyMusic = [EasyNote]

makeNote :: EasyNote -> Music
```

Näiden kahden avulla voidaankin kuvata helposti esimerkiksi tälläinen kolmen nuotin ihme ja ulos saadaan Music-tyyppiä:

```
makeNote(10,2):+:makeNote(8,2):+:makeNote(5,2)
--tulostaa:
Note (A,0) (2 % 1) [] :+: (Note (G,0) (2 % 1) [] :+: Note (E,0) (2 % 1) [])
```

9 Yhteenveto

Tässä paperissa keskityttiin lähinnä MIDIn käyttöön Haskellin kautta Haskoren avulla. MIDI on tarkoitettu lähinnä musiikin kuvaamiseen, joten jotta Haskoresta saa kaiken tehokkaasti irti, tulee musiikin teorian perusteiden olla hallussa.

Musiikin kuvaamiseen on varmasti tehokkaampia ja graafisempia työkaluja kuin Haskore, mutta ne eivät ole aina käytettävissä ja voivat olla maksullisia. Tällaisenaan Haskore ja sen päälle rakennettu HasChorus tarjoavat

erittäin laajat mahdollisuudet kuvata musiikkia ja kääntää sen MIDIksi, mikäli vain uskallusta ja osaamista riittää.

Haskoren opiskeluun paras lähde on luonnollisesti Paul Hudakin Haskore Music Tutorial[5], joka kertoo yksiselitteisesti tutoriaalin omaisesti kuinka Haskore on rakentunut ja kuinka sitä käytetään.

Lähteet

[1] Edward Cox, *Music on the Internet*, luku: How the Computer Became a Music System, saatavilla osoitteesta:`http://www.edcox.net/research/ma/ma1-soundhistory.asp`, 1998, viitattu 23.4.2008.

[2] Haskellin äänikirjastoja, *Hackage: Sound libraries*, Saativalla osoitteesta: `http://hackage.haskell.org/packages/archive/pkg-list.html#cat:Sound`, viitattu 28.4.2008.

[3] Henning Thielemann, *Audio processing in Haskell*, Center of Industrial Mathematics,University of Bremen, Bremen, Germany, 2004.

[4] Paul Hudak, *The Haskell School of Expression*, Yale University. Department of Computer Science, 2000, s. 287–320.

[5] Paul Hudak, *Haskore Music Tutorial*, Yale University. Department of Computer Science, Helmikuu 14, 1997, (Revised February 2000).

[6] Haskell Wiki, *Applications and libraries, Music and sound*, saatavilla osoitteesta: `http://www.haskell.org/haskellwiki/Applications_and_libraries/Music_and_sound`,viitattu 20.4.2008

[7] Haskoren uusin, epävakaa versio, *Haskore Music System: a revised and extended version of Haskore*, saatavilla osoitteesta: `http://darcs.haskell.org/haskore/`, viitattu 23.4.2008.

[8] MIDI Manufacters association *General MIDI Instrument patch map*, saatavilla osoitteesta: `http://www.midi.org/about-midi/gm/gm1sound.shtml`, viitattu 20.4.2008.

[9] Martin Schwenke, *HasChorus Readme,saatavilla osoitteesta: http://meltin.net/hacks/haskell/haschorus/*, 2001, viitattu 18.4.2008