Musik und Sprache: Baumbasierte Musikgenerierung

Maximilian Marx

TU Dresden

2010-07-01

Inhalt

Motivation

- Musik als formale Sprache?

Motivation

- Musik als formale Sprache?
- Erzeugbare Strukturen?

Phrasenstrukturgrammatik

Definition (Phrasenstrukturgrammatik)

$G = (N, \Sigma, P, S)$ Phrasenstrukturgrammatik:

N Nichtterminalsymbole

Phrasenstrukturgrammatik

Definition (Phrasenstrukturgrammatik)

$G = (N, \Sigma, P, S)$ Phrasenstrukturgrammatik:

- N Nichtterminalsymbole
- Σ Terminalsymbole

Phrasenstrukturgrammatik

Definition (Phrasenstrukturgrammatik)

$G = (N, \Sigma, P, S)$ Phrasenstrukturgrammatik:

- N Nichtterminalsymbole
- Σ Terminalsymbole
- P Produktionen

$$P \subseteq \{(N \cup \Sigma)^* N (N \cup \Sigma)^* \to (N \cup \Sigma)^*\}$$

Phrasenstrukturgrammatik

Definition (Phrasenstrukturgrammatik)

$G = (N, \Sigma, P, S)$ Phrasenstrukturgrammatik:

- N Nichtterminalsymbole
- Σ Terminalsymbole
- P Produktionen

$$P \subseteq \{(N \cup \Sigma)^* N (N \cup \Sigma)^* \to (N \cup \Sigma)^*\}$$

- S Startsymbol

Motivation für Transformationsgrammatiken

- Nur lokale Kontextabhängigkeit in Produktionen:

Motivation für Transformationsgrammatiken

- Nur lokale Kontextabhängigkeit in Produktionen: Regeln für Inversion?

Motivation für Transformationsgrammatiken

- Nur lokale Kontextabhängigkeit in Produktionen: Regeln für Inversion?
- Tiefenstruktur

Motivation für Transformationsgrammatiken

- Nur lokale Kontextabhängigkeit in Produktionen: Regeln für Inversion?
- Tiefenstruktur
- Oberflächenstruktur

Transformationsgrammatik

Definition (Transformationsgrammatik)

$TG = (G, T)$ Transformationsgrammatik:

G Phrasenstrukturgrammatik

Transformationsgrammatik

Definition (Transformationsgrammatik)

$TG = (G, T)$ Transformationsgrammatik:

- G Phrasenstrukturgrammatik
- T Transformationsregeln

Inhalt

Rangbehaftete Alphabete

Definition (Rangbehaftetes Alphabet)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet:

S Symbole

Rangbehaftete Alphabete

Definition (Rangbehaftetes Alphabet)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet:

S Symbole

ar Aritätsfunktion:

$$\mathrm{ar} : S \to \mathbb{N}$$

Rangbehaftete Alphabete

Definition (Rangbehaftetes Alphabet)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet:

S Symbole

ar Aritätsfunktion:

$$\mathrm{ar} : S \to \mathbb{N}$$

Schreibweise:

$$\Sigma_n := \{s \in S; \mathrm{ar}(s) = n\}$$

Terme

Definition (Term)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet,
X mit $X \cap S = \varnothing$ Menge.

Terme

Definition (Term)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet,
X mit $X \cap S = \varnothing$ Menge.

- $x \in X$ ist Term

Terme

Definition (Term)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet,
X mit $X \cap S = \varnothing$ Menge.

- $x \in X$ ist Term
- $s[] := s \in \Sigma_0$ ist Term

Terme

Definition (Term)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet,
X mit $X \cap S = \varnothing$ Menge.

- $x \in X$ ist Term
- $s[] := s \in \Sigma_0$ ist Term
- $s \in \Sigma_n$, $n \geq 1$, $t_1, \ldots, t_n$ Terme:
  $s[t_1 \cdots t_n]$ Term

Terme

Definition (Term)

$\Sigma = (S, \mathrm{ar})$ rangbehaftetes Alphabet,
X mit $X \cap S = \varnothing$ Menge.

- $x \in X$ ist Term
- $s[] := s \in \Sigma_0$ ist Term
- $s \in \Sigma_n$, $n \geq 1$, $t_1, \ldots, t_n$ Terme:
  $s[t_1 \cdots t_n]$ Term

Definition (Termalgebra)

$T_\Sigma(X) := \{t; t \text{ Term}\}$ Termalgebra (Signatur Σ)

Bäume

Definition (Baum)

Darstellung von Termen:

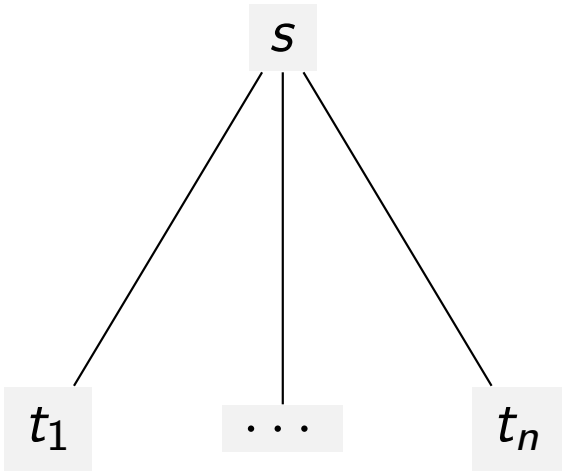
$$s[t_1 \cdots t_n] \qquad \text{Term}$$

Bäume

Definition (Baum)

Darstellung von Termen:

$s[t_1 \cdots t_n]$ Term



Reguläre Baumgrammatiken

Definition (Reguläre Baumgrammatik)

$G = (N, \Sigma', P, S)$ reguläre Baumgrammatik (über Σ), falls:

- G Phrasenstrukturgrammatik

Reguläre Baumgrammatiken

Definition (Reguläre Baumgrammatik)

$G = (N, \Sigma', P, S)$ reguläre Baumgrammatik (über Σ), falls:

- G Phrasenstrukturgrammatik
- $\Sigma = (\Sigma', \mathrm{ar})$ rangbehaftetes Alphabet

Reguläre Baumgrammatiken

Definition (Reguläre Baumgrammatik)

$G = (N, \Sigma', P, S)$ reguläre Baumgrammatik (über Σ), falls:

- G Phrasenstrukturgrammatik
- $\Sigma = (\Sigma', \mathrm{ar})$ rangbehaftetes Alphabet
- $N \cap \Sigma' = \varnothing$

Reguläre Baumgrammatiken

Definition (Reguläre Baumgrammatik)

$G = (N, \Sigma', P, S)$ reguläre Baumgrammatik (über Σ), falls:

- G Phrasenstrukturgrammatik
- $\Sigma = (\Sigma', \mathrm{ar})$ rangbehaftetes Alphabet
- $N \cap \Sigma' = \varnothing$
- $S \in N$

Reguläre Baumgrammatiken

Definition (Reguläre Baumgrammatik)

$G = (N, \Sigma', P, S)$ reguläre Baumgrammatik (über Σ), falls:

- G Phrasenstrukturgrammatik
- $\Sigma = (\Sigma', \mathrm{ar})$ rangbehaftetes Alphabet
- $N \cap \Sigma' = \varnothing$
- $S \in N$
- $P \subseteq \{N \to T_\Sigma(N)\}$

Reguläre Baumgrammatiken

Definition (Reguläre Baumgrammatik)

$G = (N, \Sigma', P, S)$ reguläre Baumgrammatik (über Σ), falls:

- G Phrasenstrukturgrammatik
- $\Sigma = (\Sigma', \mathrm{ar})$ rangbehaftetes Alphabet
- $N \cap \Sigma' = \varnothing$
- $S \in N$
- $P \subseteq \{N \to T_\Sigma(N)\}$

$L(G)$ Menge von G erzeugter Bäume

Motivation für Baumtransduktoren

- Realisierung von Transformationen

Motivation für Baumtransduktoren

- Realisierung von Transformationen
- Zustandsautomaten

Motivation für Baumtransduktoren

- Realisierung von Transformationen
- Zustandsautomaten
- Benutzereingaben erfordern Toleranz

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

Σ endliches rangbehaftetes Eingabealphabet

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

- Σ endliches rangbehaftetes Eingabealphabet
- Ω endliches rangbehaftetes Ausgabealphabet

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

- Σ endliches rangbehaftetes Eingabealphabet
- Ω endliches rangbehaftetes Ausgabealphabet
- Q rangbehaftetes Zustandsalphabet:

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

- Σ endliches rangbehaftetes Eingabealphabet
- Ω endliches rangbehaftetes Ausgabealphabet
- Q rangbehaftetes Zustandsalphabet:
  - $Q_0 = \varnothing$

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

- Σ endliches rangbehaftetes Eingabealphabet
- Ω endliches rangbehaftetes Ausgabealphabet
- Q rangbehaftetes Zustandsalphabet:
  - $Q_0 = \varnothing$
  - $Q \cap \Sigma = \varnothing = Q \cap \Omega$

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

- Σ endliches rangbehaftetes Eingabealphabet
- Ω endliches rangbehaftetes Ausgabealphabet
- Q rangbehaftetes Zustandsalphabet:
  - $Q_0 = \varnothing$
  - $Q \cap \Sigma = \varnothing = Q \cap \Omega$
- Q_t tolerante Zustände: $Q_t \subseteq Q$

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

- Σ endliches rangbehaftetes Eingabealphabet
- Ω endliches rangbehaftetes Ausgabealphabet
- Q rangbehaftetes Zustandsalphabet:
  - $Q_0 = \varnothing$
  - $Q \cap \Sigma = \varnothing = Q \cap \Omega$
- Q_t tolerante Zustände: $Q_t \subseteq Q$
- R Transformationsregeln

Baumtransduktoren

Definition (Toleranter Makro-Baumtransduktor)

$T = (\Sigma, \Omega, Q, Q_t, R, q_0)$ toleranter Makro-Baumtransduktor:

- Σ endliches rangbehaftetes Eingabealphabet
- Ω endliches rangbehaftetes Ausgabealphabet
- Q rangbehaftetes Zustandsalphabet:
  - $Q_0 = \varnothing$
  - $Q \cap \Sigma = \varnothing = Q \cap \Omega$
- Q_t tolerante Zustände: $Q_t \subseteq Q$
- R Transformationsregeln
- q_0 Initialzustand

Toleranter Makro-Baumtransduktor II

Definition

$$R \subseteq \{q[f[x_1, \ldots, x_k], y_1, \ldots, y_m] \to t;\ q \in Q_{m+1}, f \in \Sigma_k, x_1, \ldots, x_k, y_1, \ldots, y_m \in X, k, m \in \mathbb{N}, t \in T\}$$

Toleranter Makro-Baumtransduktor II

Definition

$$R \subseteq \{q[f[x_1, \ldots, x_k], y_1, \ldots, y_m] \to t;\ q \in Q_{m+1}, f \in \Sigma_k, x_1, \ldots, x_k, y_1, \ldots, y_m \in X, k, m \in \mathbb{N}, t \in T\}$$

- $y_i \in T$ $(i \in \{1, \ldots, m\})$

Toleranter Makro-Baumtransduktor II

Definition

$$R \subseteq \{q[f[x_1, \ldots, x_k], y_1, \ldots, y_m] \to t;\ q \in Q_{m+1}, f \in \Sigma_k, x_1, \ldots, x_k, y_1, \ldots, y_m \in X, k, m \in \mathbb{N}, t \in T\}$$

- $y_i \in T$ $(i \in \{1, \ldots, m\})$
- $g \in \Omega_l,\ t_1, \ldots, t_l \in T$: $g[t_1, \ldots, t_l] \in T$

Toleranter Makro-Baumtransduktor II

Definition

$$R \subseteq \{q[f[x_1, \ldots, x_k], y_1, \ldots, y_m] \to t;\ q \in Q_{m+1}, f \in \Sigma_k, x_1, \ldots, x_k, y_1, \ldots, y_m \in X, k, m \in \mathbb{N}, t \in T\}$$

- $y_i \in T$ $(i \in \{1, \ldots, m\})$
- $g \in \Omega_l,\ t_1, \ldots, t_l \in T$:
  $g[t_1, \ldots, t_l] \in T$
- $q' \in Q_{l+1},\ i \in \{1, \ldots, k\},\ t_1, \ldots, t_l \in T$:
  $q'[x_i, t_1, \ldots, t_l] \in T$

Inhalt

Musikstücke

Definition (Note)

$n = (t, l) \in \mathbb{Z} \times \mathbb{R}^+$ Note:

t Ton

Musikstücke

Definition (Note)

$n = (t, l) \in \mathbb{Z} \times \mathbb{R}^+$ Note:

- t Ton
- l Länge

Musikstücke

Definition (Note)

$n = (t, l) \in \mathbb{Z} \times \mathbb{R}^+$ Note:

t Ton

l Länge

$\mathcal{N}$ Menge aller Noten

Musikstücke

Definition (Note)

$n = (t, l) \in \mathbb{Z} \times \mathbb{R}^+$ Note:

t Ton

l Länge

$\mathcal{N}$ Menge aller Noten

Definition (Musikstück)

(N, L) Musikstück:

N Gespielte Noten: $N \subseteq \mathcal{N} \times \mathbb{R}_0^+$

Musikstücke

Definition (Note)

$n = (t, l) \in \mathbb{Z} \times \mathbb{R}^+$ Note:

t Ton

l Länge

$\mathcal{N}$ Menge aller Noten

Definition (Musikstück)

(N, L) Musikstück:

N Gespielte Noten: $N \subseteq \mathcal{N} \times \mathbb{R}_0^+$

L Länge $L \in \mathbb{R}^+$

Musikstücke

Definition (Note)

$n = (t, l) \in \mathbb{Z} \times \mathbb{R}^+$ Note:

t Ton

l Länge

$\mathcal{N}$ Menge aller Noten

Definition (Musikstück)

(N, L) Musikstück:

N Gespielte Noten: $N \subseteq \mathcal{N} \times \mathbb{R}_0^+$

L Länge $L \in \mathbb{R}^+$

$$\forall((t, l), s) \in N : L \geq l + s$$

Musik-Algebra M

Musikalgebra M: Mehrsortige universelle Algebra mit Sorten:

$\mathcal{P}$ Menge aller Musikstücke

Musik-Algebra M

Musikalgebra M: Mehrsortige universelle Algebra mit Sorten:

- $\mathcal{P}$ Menge aller Musikstücke
- $\mathbb{Z}$ Töne

Musik-Algebra M

Musikalgebra M: Mehrsortige universelle Algebra mit Sorten:

- $\mathcal{P}$ Menge aller Musikstücke
- $\mathbb{Z}$ Töne
- $\mathbb{R}^+$ Zeit

Operationen

Operationen auf M:

- $+, - : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$

Operationen

Operationen auf M:

- $+, - : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$
- $+, \cdot, \mathsf{max}, \mathsf{min} : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

Operationen

Operationen auf M:

- $+, - : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$
- $+, \cdot, \mathsf{max}, \mathsf{min} : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

Schreibweise: $t_1 \circ t_2 := \circ[t_1, t_2]$

Operationen

Operationen auf M:

- $+, - : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$
- $+, \cdot, \max, \min : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

Schreibweise: $t_1 \circ t_2 := \circ[t_1, t_2]$

N Menge gespielter Noten

- $N[\![(n, s) \mapsto f(n, s)]\!] := \{f(n, s); (n, s) \in N\}$

Operationen

Operationen auf M:

- $+, - : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$
- $+, \cdot, \max, \min : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

Schreibweise: $t_1 \circ t_2 := \circ[t_1, t_2]$

N Menge gespielter Noten

- $N[\![(n, s) \mapsto f(n, s)]\!] := \{f(n, s); (n, s) \in N\}$
- $N[\![n \mapsto f(n)]\!] := \{(f(n), s); (n, s) \in N\}$

Operationen

Operationen auf M:

- $+, - : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$
- $+, \cdot, \max, \min : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

Schreibweise: $t_1 \circ t_2 := \circ[t_1, t_2]$

N Menge gespielter Noten

- $N[\![(n, s) \mapsto f(n, s)]\!] := \{f(n, s); (n, s) \in N\}$
- $N[\![n \mapsto f(n)]\!] := \{(f(n), s); (n, s) \in N\}$

Erinnerung: $n = (t_n, l_n)$

Operationen II

$P = (N, L)$, $P' = (N', L')$ Musikstücke:

LENGTH(P) Länge

Operationen II

$P = (N, L)$, $P' = (N', L')$ Musikstücke:

LENGTH(P) Länge

HIGHEST(P) Höchster Ton am Ende von P:

Operationen II

$P = (N, L)$, $P' = (N', L')$ Musikstücke:

LENGTH(P) Länge

HIGHEST(P) Höchster Ton am Ende von P:

$$\text{HIGHEST}(P) = \begin{cases} \max\{t_n; (n, s) \in N, s + l_n = q\} & N \neq \varnothing \\ 0 & \text{sonst} \end{cases}$$

$$q := \max\{s + l_n; (n, s) \in N\}$$

Operationen II

$P = (N, L)$, $P' = (N', L')$ Musikstücke:

LENGTH(P) Länge

HIGHEST(P) Höchster Ton am Ende von P:

$$\text{HIGHEST}(P) = \begin{cases} \max\{t_n; (n, s) \in N, s + l_n = q\} & N \neq \varnothing \\ 0 & \text{sonst} \end{cases}$$

$$q := \max\{s + l_n; (n, s) \in N\}$$

SCALE(P, a) Skalierung um $a \in \mathbb{R}^+$:

Operationen II

$P = (N, L)$, $P' = (N', L')$ Musikstücke:

LENGTH(P) Länge

HIGHEST(P) Höchster Ton am Ende von P:

$$\text{HIGHEST}(P) = \begin{cases} \max\{t_n; (n, s) \in N, s + l_n = q\} & N \neq \varnothing \\ 0 & \text{sonst} \end{cases}$$

$$q := \max\{s + l_n; (n, s) \in N\}$$

SCALE(P, a) Skalierung um $a \in \mathbb{R}^+$:

$$\text{SCALE}(P, a) := (N[\![(n, s) \mapsto ((t_n, a \cdot l_n), a \cdot s)]\!], a \cdot L)$$

Operationen III

INV(P) Inversion:

Operationen III

$\mathrm{INV}(P)$ Inversion:

$$\mathrm{INV}(P) := (N[\![n \mapsto (-t_n, l_n)]\!], L)$$

Operationen III

$\mathrm{INV}(P)$ Inversion:

$$\mathrm{INV}(P) := (N[\![n \mapsto (-t_n, l_n)]\!], L)$$

$\mathrm{RAISE}(P, t)$ Transposition:

Operationen III

$\mathrm{INV}(P)$ Inversion:

$$\mathrm{INV}(P) := (N[\![n \mapsto (-t_n, l_n)]\!], L)$$

$\mathrm{RAISE}(P, t)$ Transposition:

$$\mathrm{RAISE}(P, t) := (N[\![n \mapsto (t_n + t, l_n)]\!], L)$$

Operationen III

$\mathrm{INV}(P)$ Inversion:

$$\mathrm{INV}(P) := (N[\![n \mapsto (-t_n, l_n)]\!], L)$$

$\mathrm{RAISE}(P, t)$ Transposition:

$$\mathrm{RAISE}(P, t) := (N[\![n \mapsto (t_n + t, l_n)]\!], L)$$

$\mathrm{BACK}(P)$ Krebs:

Operationen III

$\mathrm{INV}(P)$ Inversion:

$$\mathrm{INV}(P) := (N[\![n \mapsto (-t_n, l_n)]\!], L)$$

$\mathrm{RAISE}(P, t)$ Transposition:

$$\mathrm{RAISE}(P, t) := (N[\![n \mapsto (t_n + t, l_n)]\!], L)$$

$\mathrm{BACK}(P)$ Krebs:

$$\mathrm{BACK}(P) := (N[\![(n, s) \mapsto (n, L - s - l_n)]\!], L)$$

Operationen III

$\mathrm{INV}(P)$ Inversion:

$$\mathrm{INV}(P) := (N[\![n \mapsto (-t_n, l_n)]\!], L)$$

$\mathrm{RAISE}(P, t)$ Transposition:

$$\mathrm{RAISE}(P, t) := (N[\![n \mapsto (t_n + t, l_n)]\!], L)$$

$\mathrm{BACK}(P)$ Krebs:

$$\mathrm{BACK}(P) := (N[\![(n, s) \mapsto (n, L - s - l_n)]\!], L)$$

$\mathrm{MUTE}(P)$ Stille:

Operationen III

$\mathrm{INV}(P)$ Inversion:

$$\mathrm{INV}(P) := (N[\![n \mapsto (-t_n, l_n)]\!], L)$$

$\mathrm{RAISE}(P, t)$ Transposition:

$$\mathrm{RAISE}(P, t) := (N[\![n \mapsto (t_n + t, l_n)]\!], L)$$

$\mathrm{BACK}(P)$ Krebs:

$$\mathrm{BACK}(P) := (N[\![(n, s) \mapsto (n, L - s - l_n)]\!], L)$$

$\mathrm{MUTE}(P)$ Stille:

$$\mathrm{MUTE}(P) := (\varnothing, L)$$

Operationen IV

OVERLAY(P, P') Überlagerung:

Operationen IV

OVERLAY(P, P') Überlagerung:

$$\text{OVERLAY}(P, P') := (N \cup N', \max(L, L'))$$

$$\text{OVERLAY}(P_1, \ldots, P_k) := (\bigcup_{i \in \{1,\ldots,k\}} N_i, \max_{i \in \{1,\ldots,k\}} L_i)$$

Operationen IV

OVERLAY(P, P') Überlagerung:

$$\text{OVERLAY}(P, P') := (N \cup N', \max(L, L'))$$

$$\text{OVERLAY}(P_1, \ldots, P_k) := (\bigcup_{i \in \{1,\ldots,k\}} N_i, \max_{i \in \{1,\ldots,k\}} L_i)$$

CONCAT(P, P') Konkatenation:

Operationen IV

OVERLAY(P, P') Überlagerung:

$$\text{OVERLAY}(P, P') := (N \cup N', \max(L, L'))$$

$$\text{OVERLAY}(P_1, \ldots, P_k) := \left(\bigcup_{i \in \{1,\ldots,k\}} N_i, \max_{i \in \{1,\ldots,k\}} L_i\right)$$

CONCAT(P, P') Konkatenation:

$$\text{CONCAT}(P, P') := (N \cup N' [\![(n, s) \mapsto (n, L + s)]\!], L + L')$$

$$\text{CONCAT}(P_1, \ldots, P_n) := \text{CONCAT}(P_1, \cdots \text{CONCAT}(P_{k-1}, P_k) \cdots)$$

Operationen V

$S \subseteq \mathbb{Z}$ endliche, nichtleere Menge von Tönen:

$\mathrm{SNAP}_S(P)$ Ausrichtung von P am nächsten Ton von S:

Operationen V

$S \subseteq \mathbb{Z}$ endliche, nichtleere Menge von Tönen:

$\mathrm{SNAP}_S(P)$ Ausrichtung von P am nächsten Ton von S:

$t \in \mathbb{Z}$ Ton:

- $\Delta(t) := \min_{s \in S} |t - s|$

Operationen V

$S \subseteq \mathbb{Z}$ endliche, nichtleere Menge von Tönen:

$\mathrm{SNAP}_S(P)$ Ausrichtung von P am nächsten Ton von S:

$t \in \mathbb{Z}$ Ton:

- $\Delta(t) := \min_{s \in S} |t - s|$
- $\alpha(t) := \begin{cases} t + \Delta(t) & t + \Delta(t) \in S \\ t - \Delta(t) & \text{sonst} \end{cases}$

Operationen V

$S \subseteq \mathbb{Z}$ endliche, nichtleere Menge von Tönen:

$\mathrm{SNAP}_S(P)$ Ausrichtung von P am nächsten Ton von S:

$t \in \mathbb{Z}$ Ton:

- $\Delta(t) := \min_{s \in S} |t - s|$
- $\alpha(t) := \begin{cases} t + \Delta(t) & t + \Delta(t) \in S \\ t - \Delta(t) & \text{sonst} \end{cases}$

$$\mathrm{SNAP}_S(P) := (N[\![n \mapsto (\alpha(t_n), l_n)]\!], L)$$

Operationen V

$S \subseteq \mathbb{Z}$ endliche, nichtleere Menge von Tönen:

$\mathrm{SNAP}_S(P)$ Ausrichtung von P am nächsten Ton von S:

$t \in \mathbb{Z}$ Ton:

- $\Delta(t) := \min_{s \in S} |t - s|$
- $\alpha(t) := \begin{cases} t + \Delta(t) & t + \Delta(t) \in S \\ t - \Delta(t) & \text{sonst} \end{cases}$

$$\mathrm{SNAP}_S(P) := (N[\![n \mapsto (\alpha(t_n), l_n)]\!], L)$$

$[\![\cdot]\!]_M$ Termauswertung in M

Inhalt

Generator

Definition (Baumbasierter Musikgenerator)

$G = (\Gamma, M)$ baumbasierter Musikgenerator:

- Γ Baumgenerator:

Generator

Definition (Baumbasierter Musikgenerator)

$G = (\Gamma, M)$ baumbasierter Musikgenerator:

- Γ Baumgenerator:
  $\Gamma = (g, td_1, \ldots, td_k)$:
  - g reguläre Baumgrammatik

Generator

Definition (Baumbasierter Musikgenerator)

$G = (\Gamma, M)$ baumbasierter Musikgenerator:

- Γ Baumgenerator:
  $\Gamma = (g, td_1, \ldots, td_k)$:
  - g reguläre Baumgrammatik
  - td_i toleranter Makro-Baumtransduktor

Generator

Definition (Baumbasierter Musikgenerator)

$G = (\Gamma, M)$ baumbasierter Musikgenerator:

- Γ Baumgenerator:
  $\Gamma = (g, td_1, \ldots, td_k)$:
  - g reguläre Baumgrammatik
  - td_i toleranter Makro-Baumtransduktor
- M Musikalgebra M

Generator

Definition (Baumbasierter Musikgenerator)

$G = (\Gamma, M)$ baumbasierter Musikgenerator:

- Γ Baumgenerator:
  $\Gamma = (g, td_1, \ldots, td_k)$:
  - g reguläre Baumgrammatik
  - td_i toleranter Makro-Baumtransduktor
- M Musikalgebra M

$L(G) = \{[\![t]\!]_M; t \in td_k(\cdots(td_1(L(g)))\cdots)\}$ erzeugte Musikstücke

Benutzereingaben

- Problem: Toleranz

Benutzereingaben

- Problem: Toleranz
- Lösung: $\Sigma = \{\mathrm{CONCAT}_2, \mathrm{OVERLAY}_2, \mathrm{MUTE}_1\}$

Benutzereingaben

- Problem: Toleranz
- Lösung: $\Sigma = \{\mathrm{CONCAT}_2, \mathrm{OVERLAY}_2, \mathrm{MUTE}_1\}$
  $\Omega = \Sigma \cup \{\mathrm{NOTE}_1\}$

Benutzereingaben

- Problem: Toleranz
- Lösung: $\Sigma = \{\mathrm{CONCAT}_2, \mathrm{OVERLAY}_2, \mathrm{MUTE}_1\}$
  $\Omega = \Sigma \cup \{\mathrm{NOTE}_1\}$

$$R = \{qf[x_1, \ldots, x_k] \to f[t_1, \ldots, t_k];\ f \in \Sigma_k,$$
$$\forall i \in \{1, \ldots, k\} : t_i \in \{q[x_i], \mathrm{NOTE}[q'[x_i]]\}\}$$

Benutzereingaben

- Problem: Toleranz
- Lösung: $\Sigma = \{\mathrm{CONCAT}_2, \mathrm{OVERLAY}_2, \mathrm{MUTE}_1\}$
  $\Omega = \Sigma \cup \{\mathrm{NOTE}_1\}$

$$R = \{qf[x_1, \ldots, x_k] \to f[t_1, \ldots, t_k];\ f \in \Sigma_k,$$
$$\forall i \in \{1, \ldots, k\} : t_i \in \left\{q[x_i], \mathrm{NOTE}[q'[x_i]]\right\}\}$$

$PRE = (\Sigma, \Omega, \{q, q'\}, \{q'\}, R, q)$ toleranter Baumtransduktor

Verzierungen

Beispiel

MORDENT toleranter Baumtransduktor mit Regel

$$S[\mathrm{NOTE}[x_1]] \to \mathrm{SCALE}[\mathrm{CONCAT}[S[x_1], \mathrm{RAISE}[S[x_1], -1], S[x_1]], \frac{1}{3} \cdot \mathrm{LENGTH}[S[x_1]]]$$

Progressionen

- Progression G

Progressionen

- Progression G
- Status: (c, c'), c, c' Akkorde

Progressionen

- Progression G
- Status: (c, c'), c, c' Akkorde
- Initialstatus (c_s, c_e): Progression $c_s \longrightarrow c_e$

Progressionen

- Progression G
- Status: (c, c'), c, c' Akkorde
- Initialstatus (c_s, c_e): Progression $c_s \to c_e$
- $(c, c')[\text{CONCAT}[x_1, x_2]] \to \text{CONCAT}[(c, c'')[x_1], (c'', c')[x_2]]$, falls $c \to c'$ via c'' in G

Progressionen

- Progression G
- Status: (c, c'), c, c' Akkorde
- Initialstatus (c_s, c_e): Progression $c_s \to c_e$
- $(c, c')[\mathrm{CONCAT}[x_1, x_2]] \to \mathrm{CONCAT}[(c, c'')[x_1], (c'', c')[x_2]]$, falls $c \to c'$ via c'' in G
- $(c, c')[\mathrm{CONCAT}[x_1, x_2]] \to \mathrm{SNAP}_{\hat{c}}[\mathrm{CONCAT}[P[x_1], P[x_2]]]$, falls $\delta(c, c') \leq 1$ in G

Progressionen

- Progression G
- Status: (c, c'), c, c' Akkorde
- Initialstatus (c_s, c_e): Progression $c_s \to c_e$
- $(c, c')[\mathrm{CONCAT}[x_1, x_2]] \to \mathrm{CONCAT}[(c, c'')[x_1], (c'', c')[x_2]]$, falls $c \to c'$ via c'' in G
- $(c, c')[\mathrm{CONCAT}[x_1, x_2]] \to \mathrm{SNAP}_{\hat{c}}[\mathrm{CONCAT}[P[x_1], P[x_2]]]$, falls $\delta(c, c') \leq 1$ in G
- P tolerant

Progressionen

- Progression G
- Status: (c, c'), c, c' Akkorde
- Initialstatus (c_s, c_e): Progression $c_s \to c_e$
- $(c, c')[\text{CONCAT}[x_1, x_2]] \to \text{CONCAT}[(c, c'')[x_1], (c'', c')[x_2]]$, falls $c \to c'$ via c'' in G
- $(c, c')[\text{CONCAT}[x_1, x_2]] \to \text{SNAP}_{\hat{c}}[\text{CONCAT}[P[x_1], P[x_2]]]$, falls $\delta(c, c') \leq 1$ in G
- P tolerant
- $\hat{c}$ Abschluss von c unter Transposition

Weitere Anwendungen

- Kanons

Weitere Anwendungen

- Kanons
- Fugenartige Strukturen

Weitere Anwendungen

- Kanons
- Fugenartige Strukturen
- ...

Literaturhinweise

- Högberg, J.: Wind in the willows – generating music by means of tree transducers. In: Farré, J., Litovsky, I., Schmitz, S. (eds.) CIAA 20055. LNCS, vol. 3845, pp. 153—162. Springer, Heidelberg (2006)

Literaturhinweise

- Högberg, J.: Wind in the willows – generating music by means of tree transducers. In: Farré, J., Litovsky, I., Schmitz, S. (eds.) CIAA 20055. LNCS, vol. 3845, pp. 153—162. Springer, Heidelberg (2006)
- Drewes, F., Högberg, J.: An Algebra for tree-based music generation. In: Bozapadilis, S., Rahonis, G. (eds.) CAI 2007. LNCS, vol. 4728, pp. 172—188. Springer, Heidelberg (2007)