Biinfinite Kontrollpolygone: $\mathbf{c}_{\mathbb{Z}} := (\mathbf{c}_i)_{i\in\mathbb{Z}} = (...\mathbf{c}_{-1}\mathbf{c}_0\mathbf{c}_1...)$

Algorithmus

Eingabe:
$\mathbf{c}^0_{\mathbb{Z}}$ ein Polygon $\subset \mathbb{R}^d$
$n \in \mathbb{N}_0$ der Grad
Ausgabe:
$\mathbf{c}^m_{\mathbb{Z}} \subset \mathbb{R}^d$

For k = 1, ..., m
  For i $\in \mathbb{Z}$ //verdoppele
    $\mathbf{d}^0_{2i} \leftarrow \mathbf{d}^0_{2i+1} \leftarrow \mathbf{c}^{k-1}_i$
  For j = 1, ..., n
    For i $\in \mathbb{Z}$ //mittele
      $\mathbf{d}^j_i \leftarrow \left(\mathbf{d}^{j-1}_{i-1} + \mathbf{d}^{j-1}_i\right)\frac{1}{2}$
  For i $\in \mathbb{Z}$ //bennen um
    $\mathbf{c}^k_i \leftarrow \mathbf{d}^n_i$

$$D = \begin{bmatrix} \ddots & & & & \\ & 1 & & & \\ & 1 & & & \\ & & 1 & & \\ & & 1 & & \\ & & & \ddots \end{bmatrix} \quad M = \frac{1}{2}\begin{bmatrix} \ddots & & & & & \\ & 1 & 1 & & & \\ & & 1 & 1 & & \\ & & & 1 & 1 & \\ & & & & & \ddots \end{bmatrix}$$

$$U^m_n \mathbf{c} = (U_n)^m \mathbf{c} = (M^n D)^m \mathbf{c}$$

Allgemein enthält U_n in den Spalten die Einträge α_i, jeweils um zwei Zeilen versetzt:

$$\alpha_i = \begin{cases} \frac{1}{2^n}\binom{n+1}{i}, & i = 0, ..., n+1 \\ 0, & sonst \end{cases}$$

Unterteilungsgleichung:

$$b_i = \sum_{k\in\mathbb{Z}} \alpha_{i-2k} c_k,\ i \in \mathbb{Z}$$

Bsp. α eines stationären Unterteilungsalgorithmus:

$$\alpha(z) := \sum_{j\in\mathbb{Z}} \alpha_j z^j$$

$$\alpha_n(z) := \frac{1}{2^n}\sum_{i=0}^{n+1}\binom{n+1}{i} z^i = \frac{1}{2^n}(1+z)^{n+1}$$

Differenzenschema

Rückwärtsdifferenzen: $\nabla\mathbf{c} := (\nabla c_i)_{i\in\mathbb{Z}}$

$$\nabla c_i := c_i - c_{i-1}$$

Symbol: $v(z) := 1 - z$

$$\nabla\mathbf{c}(z) = v(z)\,c(z)$$

Differenzenpolygone: $b_\nabla(z) = (1-z)\,b(z)$

$$= (1-z)\,\alpha(z)\,\frac{c_\nabla(z^2)}{(1-z^2)}$$

$$= \frac{\alpha(z)}{1+z} c_\nabla(z^2)$$

Bem.: Das Differenzenschema zu $\alpha(z)$ existiert nur, wenn gilt: $\alpha(-1) = \sum_{i\in\mathbb{Z}} \alpha_{2i} - \sum_{i\in\mathbb{Z}} \alpha_{2i+1} = 0$

Bem.: Es muss $\sum_{i\in\mathbb{Z}} \alpha_{2i} = \sum_{i\in\mathbb{Z}} \alpha_{2i+1} = 1$ gelten, damit die unterteilten Polygone gegen eine Kurve konvergieren.

Allgemein

Regelmäßiges biinfinites Kontrollnetz C und das unterteilte Netz B mit den Unterteilungsmatrizen U und V:

$$C := [\mathbf{c}_{ij}]_{i,j\in\mathbb{Z}} = [\mathbf{c}_{\mathbf{i}}]_{\mathbf{i}\in\mathbb{Z}^2} =: \mathbf{c}_{\mathbb{Z}^2}$$
$$B := \mathbf{b}_{\mathbb{Z}^2} := UCV^t$$
(U, V) heißt Tempus

Bem.: Wenn U, V konvergente Kurvenunterteilungsalgorithmen, dann konvergiert $U^k C\left(V^t\right)^k$ gegen eine Fläche.

Masken

Wenn $\Gamma := \gamma_{\mathbb{Z}^2}$ biinfinite Matrix. Also

$\gamma(x, y) := [...x^{-1}x^0x^1...]\Gamma[...y^{-1}y^0y^1...]^t$
$\Gamma_{\mathbf{i}} := [\gamma_{\mathbf{i}-2\mathbf{j}}]_{\mathbf{j}\in\mathbb{Z}^2}$ heißen Masken für $\mathbf{i} = (0,0), (1,0), (0,1), (1,1)$

Symbole

$$C = \mathbf{c}_{\mathbb{Z}^2} \Rightarrow \mathbf{c}(\mathbf{x}) := \mathbf{c}(x, y) := \sum_{i\in\mathbb{Z}}\sum_{j\in\mathbb{Z}} \mathbf{c}_{ij} x^i y^j := \sum_{\mathbf{i}\in\mathbb{Z}^2} \mathbf{c}_{\mathbf{i}}\mathbf{x}^{\mathbf{i}}$$

$$B := UCV^t \Rightarrow \mathbf{b}(x, y) := \alpha(x)\,\mathbf{c}\left(x^2, y^2\right)\beta(y)$$
$$(U, V) \Rightarrow \gamma(x, y) := \alpha(x)\,\beta(y) \text{ (Symbol des Tempus)}$$

Unterteilungsgleichung: $\mathbf{b}(\mathbf{x}) = \gamma(\mathbf{x})\,\mathbf{c}\left(\mathbf{x}^2\right)$ bzw. komponentenweise: $\mathbf{b}_{\mathbf{i}} = \sum_{\mathbf{j}\in\mathbb{Z}^2} \gamma_{\mathbf{i}-2\mathbf{j}}\mathbf{c}_{\mathbf{j}}$

Eingabe:
  t = $t_1...t_n$
  s = $s_1...s_m$ // Suchtext
Ausgabe:
  kleinstes i mit $t_{i+1}...t_{i+m} = s$

i <- 0
Solange i $\leq$ n − m
  // vgl. s mit $t_{i+1}...t_{i+m}$
  j <- m
  Solange j > 0 und $s_j = t_{i+j}$
    j <- j − 1
  Falls j = 0
    gib i aus; stop
  sonst i <- i + 1

Vorkommensfunktion:

$$v : A \to 0, ..., m$$
$$a \mapsto v(a),$$
$$v(a) := min\{\ k | a \notin s_{k+1}...s_m\ und\ (a = s_k \vee k = 0)\}$$

Def.

Ein Präfix eines Wortes w, das zugleich ein Suffix von w ist, heißt *Präsuffix*. $geps(w)$ ist das größte echte Präsuffix von w.
$w_j := s_{j+i}...s_m$.

$$\gamma(j) := |geps(w_j)|$$

Suffixfunkion σ wird dargestellt:

$$\sigma(j) = min(\{k \in \{1, ..., j\} |\, \gamma(j-k) = m - j\} \cup \{m - \gamma(0)\})$$

Änderungen in „Naive Suche“

Sei $t_{i+j} \neq s_j$ und $t_{i+j+1}...t_{i+m} = s_{j+1}...s_m$.
Dann ist $v := v(t_{i+j}) \neq j$ und

- falls $v < j$, kann i um $j - v$ erhöht werden
- falls $v > j$, kann i um $m - v + 1$ erhöht werden

Bem.

Für kleines m und großes Alphabet A nun Laufzeit $O(\frac{n}{m})$

Eingabe:
  $\gamma(0\ ...\ m)$
Ausgabe:
  $\sigma(1\ ...\ m)$

For j = 1, ..., m
  σ(j) <- m − $gamma(0)$
For i = 0, ..., m−1
  k <- m − γ(i) − i
  j <- m − γ(i)
  // Es gilt jetzt γ(j−k) = m − j
  falls σ(j) > k
    dann σ(j) <- k

// wie in "Naiver Suche", letzte Zeile
// durch folgendes ersetzt
v <- v(t_{i+j})
falls v < j
  i <- i + max{j−v, σ(j)}
sonst
  i <- i + max{m−v+1, σ(j)}