Softwareprojektpraktikum Maschinelle Übersetzung

Matthias Huck, Daniel Stein
{huck,stein}@i6.informatik.rwth-aachen.de

Besprechung 5. Aufgabe 24. Juni 2010

Human Language Technology and Pattern Recognition
Lehrstuhl für Informatik 6
Computer Science Department
RWTH Aachen University, Germany

Contents

1 Maximum Entropy

Motivation:

- **möglichst viele Wissensquellen hinzunehmen**
- **aber: nicht alle werden gleich zuverlässig sein**
- **Ziel: Gewichtung der einzelnen Modelle durch Skalierungsfaktoren**

Mathematisch:

$$p(e_1^I|f_1^J) = \frac{\exp\left(\sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J)\right)}{\sum_{\tilde{e}_1^I} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(\tilde{e}_1^I, f_1^J)\right)}$$

- **für Skalierungsfaktoren λ_m und Funktionen $h_m(e_1^I, f_1^J)$**

Einfache Zusatzmodelle

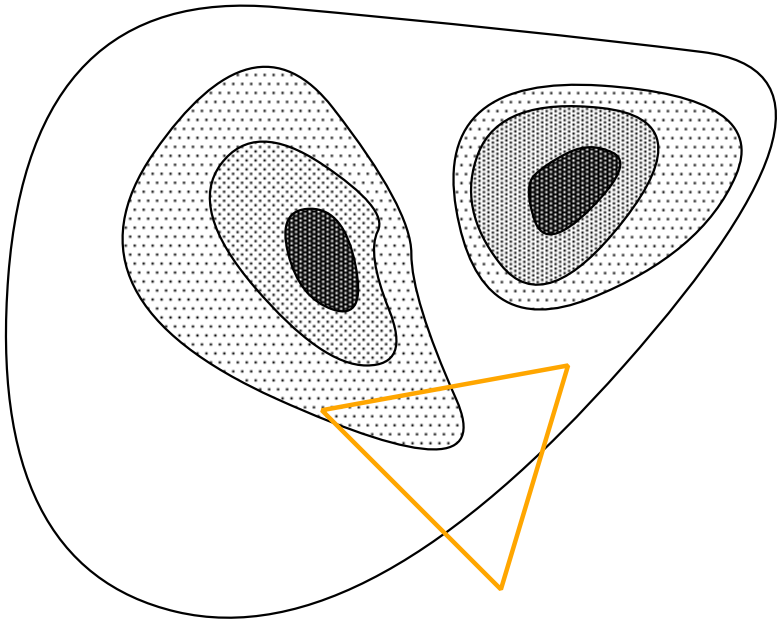
- **Anzahl der Wörter auf Target-Seite**
- **Anzahl der bei der Übersetzung benutzten Phrasen**
- **Source-Target Ratio**
- **Count-Vektoren (z.B. Phrasenvorkommen >1, >2, >5)**
- **Signifikanztests**
- **...**

2 Downhill-Simplex-Verfahren

Wie optimiert man nun die verschiedenen Skalierungsfaktoren?

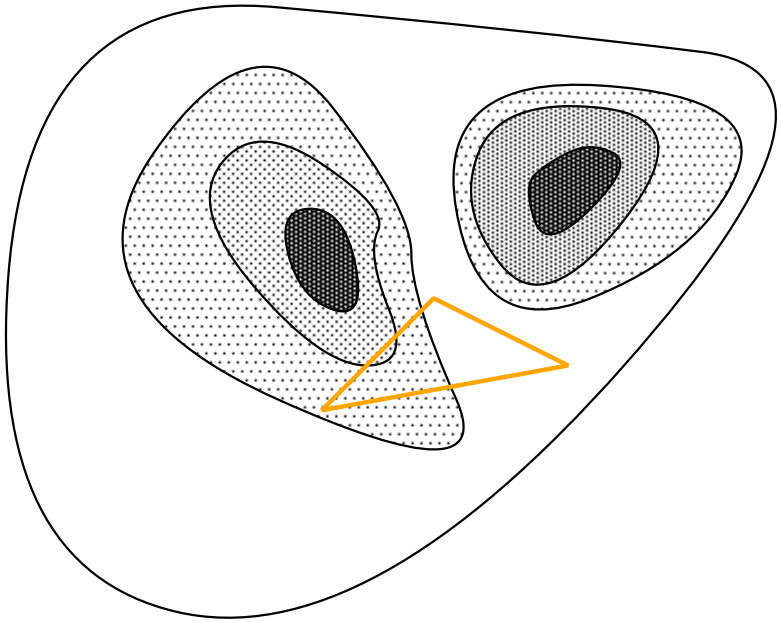
- **Downhill-Simplex**
  - **Hillclimbing-Verfahren**
  - **langsame Konvergierung, aber relativ robust**
  - **Aufstellen eines $N + 1$-Dimensionalen Vielkants (Simplex)**
  - **Schrittweise Verbesserung des schlechtesten Punktes**
- **zu verbessernde Kosten: z.B. BLEU**

Beispiel



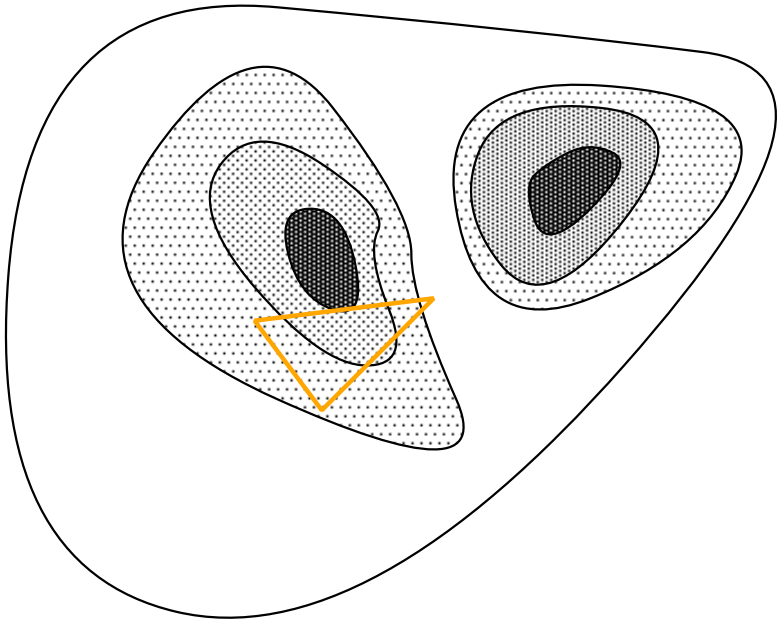
Beispielverlauf einer Simplex-Optimierung

Beispiel



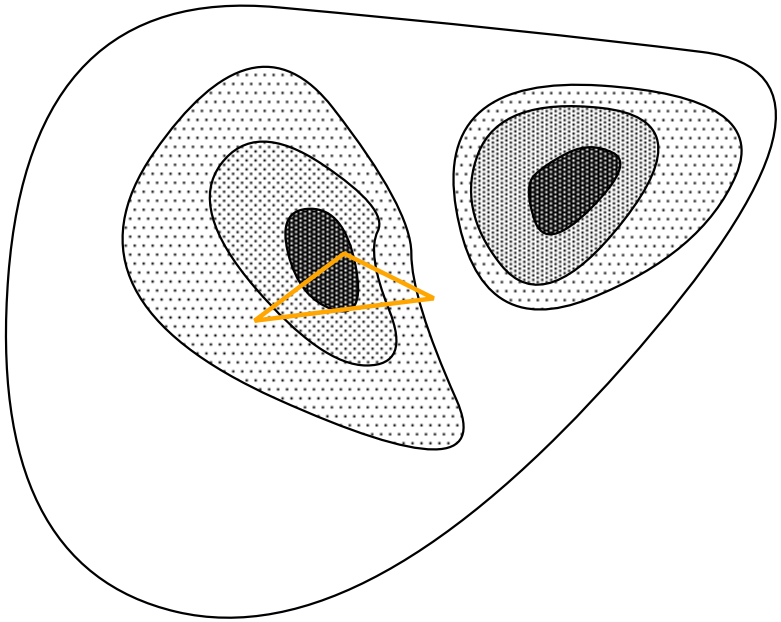
Beispielverlauf einer Simplex-Optimierung

Beispiel



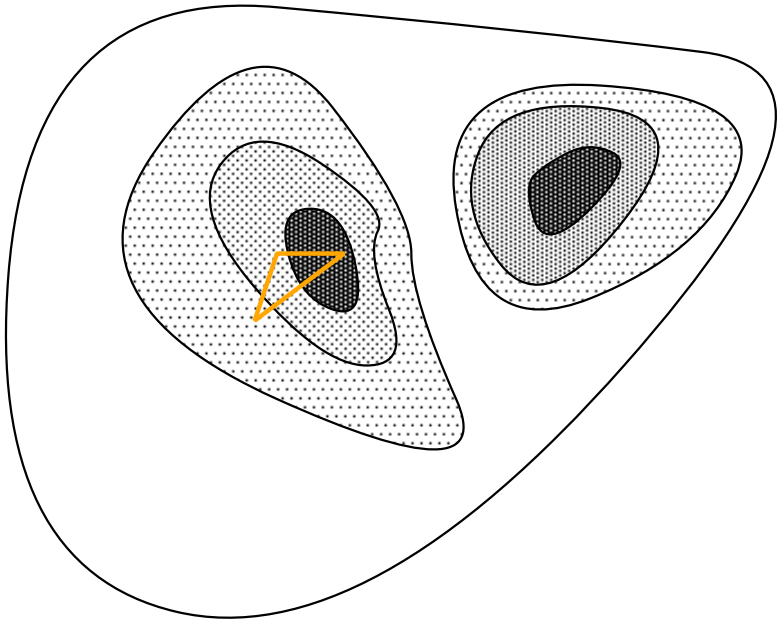
Beispielverlauf einer Simplex-Optimierung

Beispiel



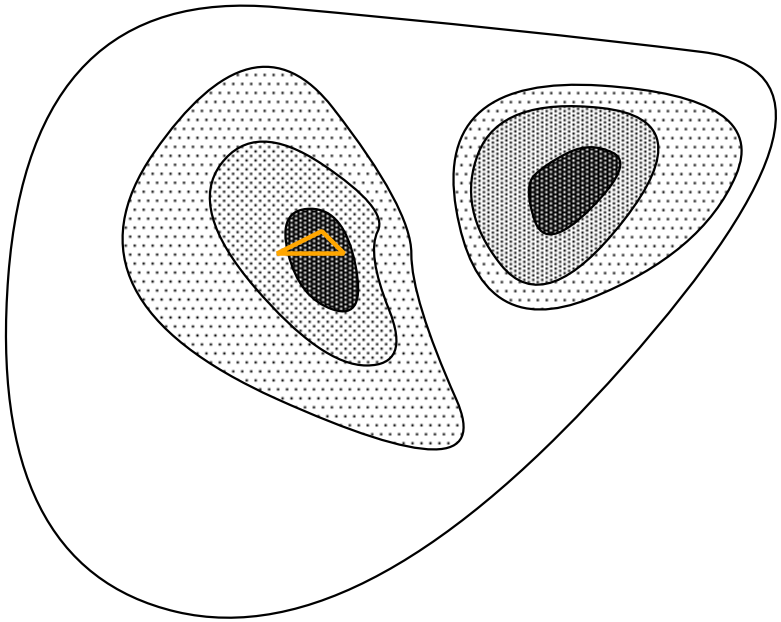
Beispielverlauf einer Simplex-Optimierung

Beispiel



Beispielverlauf einer Simplex-Optimierung

Beispiel



Beispielverlauf einer Simplex-Optimierung

Algorithmus

Initialisierung:

- **Simplex aufspannen: $N + 1$ Werte berechnen**

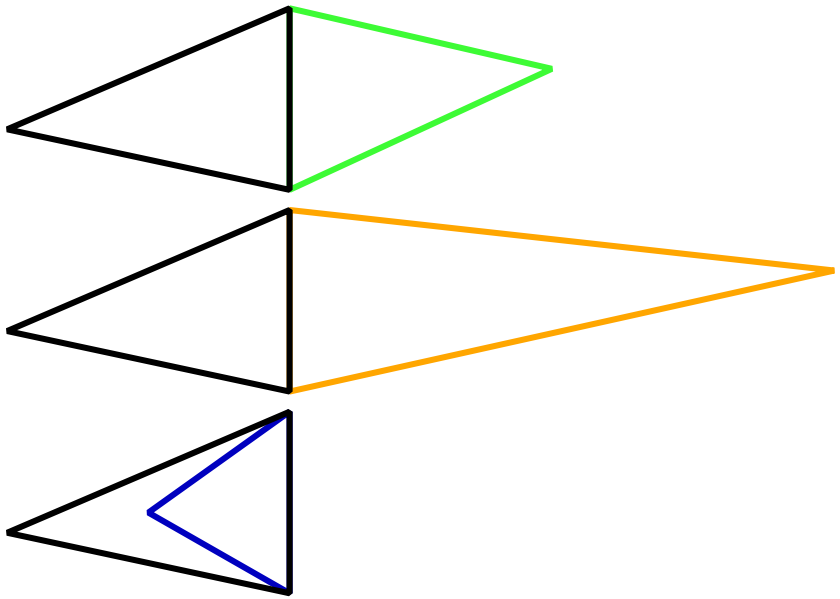
In jeder Schleife:

- **Werte sortieren, Schwerpunkt berechnen**
- **versuchen, den schwächsten Punkt zu verbessern**
- **im Notfall den Simplex zusammenziehen**

Schwächsten Punkt verbessern

- **Berechnung des Simplex-Schwerpunktes** $x_0 = \frac{\sum_{n=1}^{N} x_i}{N}$
- **Verlagerung des schlechtesten Wertes**

- **durch Reflektion:**
  $x_r = x_0 + \alpha(x_0 - x_{worst})$
- **durch Erweiterung:**
  $x_e = x_0 + \gamma(x_0 - x_{worst})$
- **durch Kontraktion:**
  $x_c = x_0 + \rho(x_0 - x_{worst})$



- **typische Werte:** $\alpha = 1.0, \gamma = 2.0, \rho = 0.5$

Downhill Simplex

Algorithm 1: Downhill Simplex

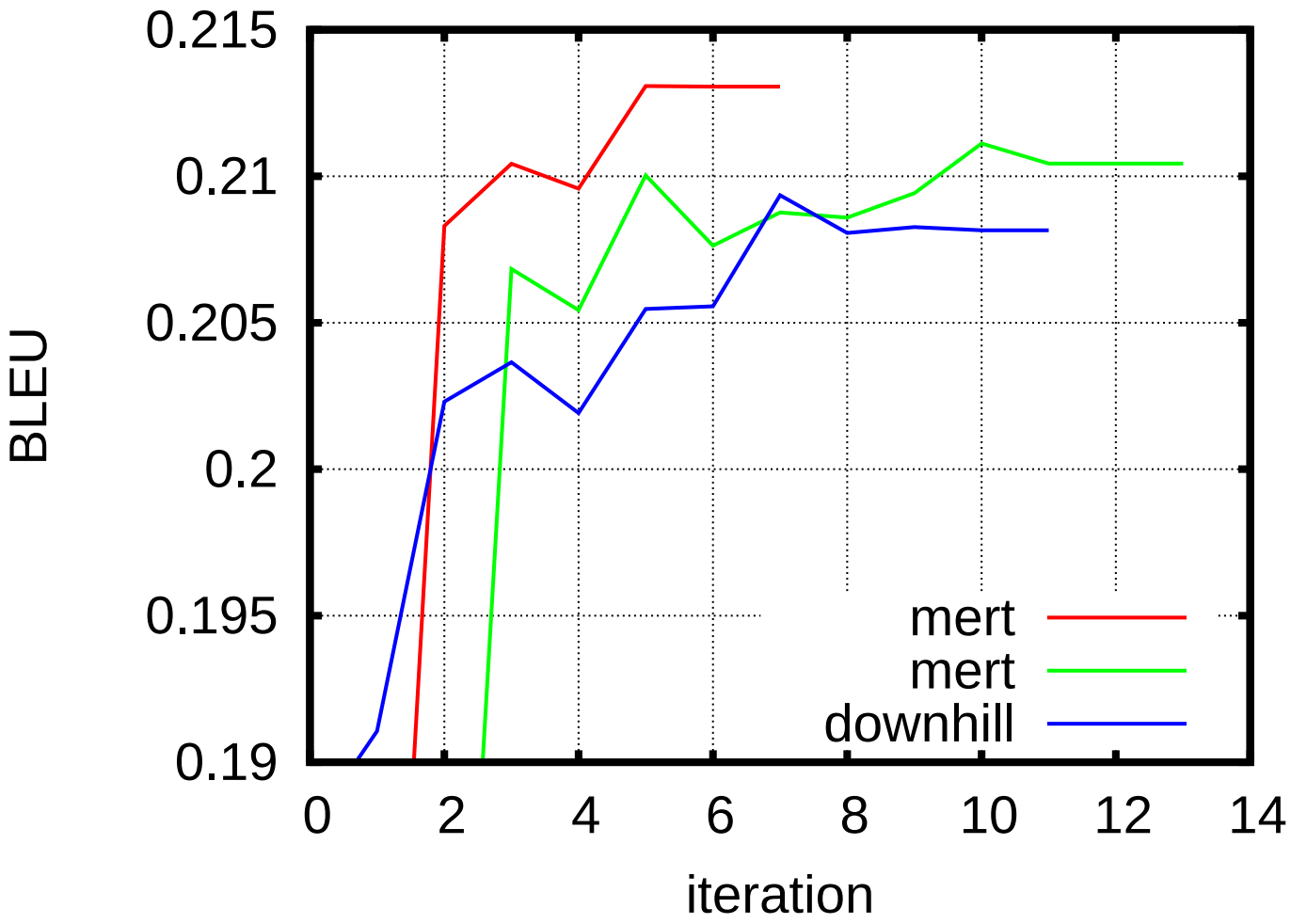
while *Optimierungskriterium nicht erreicht* **do**
- SORTIERE Simplexpunkte nach Wert: $f(x_1) \leq f(x_2) \leq \cdots \leq f(x_{N+1})$;
- BERECHNE Schwerpunkt x_0 aller Punkte ausser x_{N+1} ;
- **if** $f(x_1) \leq f(x_r) \leq f(x_N)$ **then**
  - REFLEKTIERE, d.h. ersetze x_{N+1} durch x_r;
- **else if** $f(x_r) < f(x_1)$ **then**
  - **if** $f(x_e) \leq f(x_r) \leq f(x_1)$ **then**
    - EXPANDIERE, d.h. ersetze x_{N+1} durch x_e;
  - **else**
    - REFLEKTIERE, d.h. ersetze x_{N+1} durch x_r;
- **else** // $f(x_r) > f(x_N)$
  - **if** $f(x_c) \leq f(x_{N+1})$ **then**
    - KOMPRIMIERE, d.h. ersetze x_{N+1} durch x_c;
  - **else**
    - KONTRAHIERE Simplex, $x_i = x_1 + \sigma(x_i - x_1) \qquad \forall i \in \{2, \ldots, N+1\}$;

3 Och's Minimum Error Rate Training (MERT)

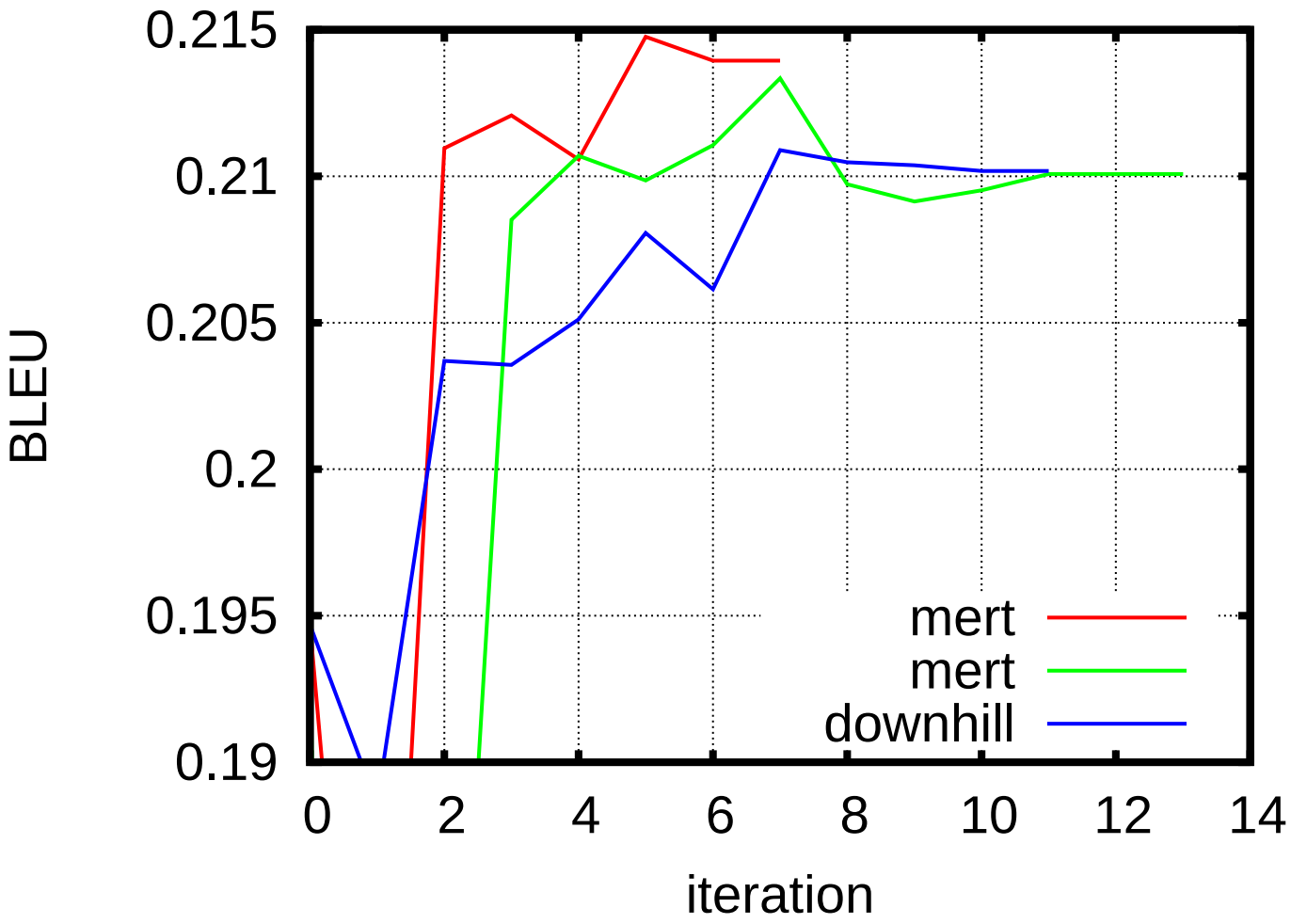
- Basiert auf der Powell's Methode
- Üblicherweise schneller und stabiler als Downhill Simplex
- Ansatz: es wird jeweils nur ein Parameter λ_k auf einmal verändert
- Idee: nicht jede Position wird berechnet, sondern nur die wesentlichen

$$\sum_m \lambda_m h_m(e_1^I, f_1^J) = \sum_{m \neq k} \lambda_m h_m(e_1^I, f_1^J) + \lambda_k h_k(e_1^I, f_1^J)$$

Results on newsdev 08

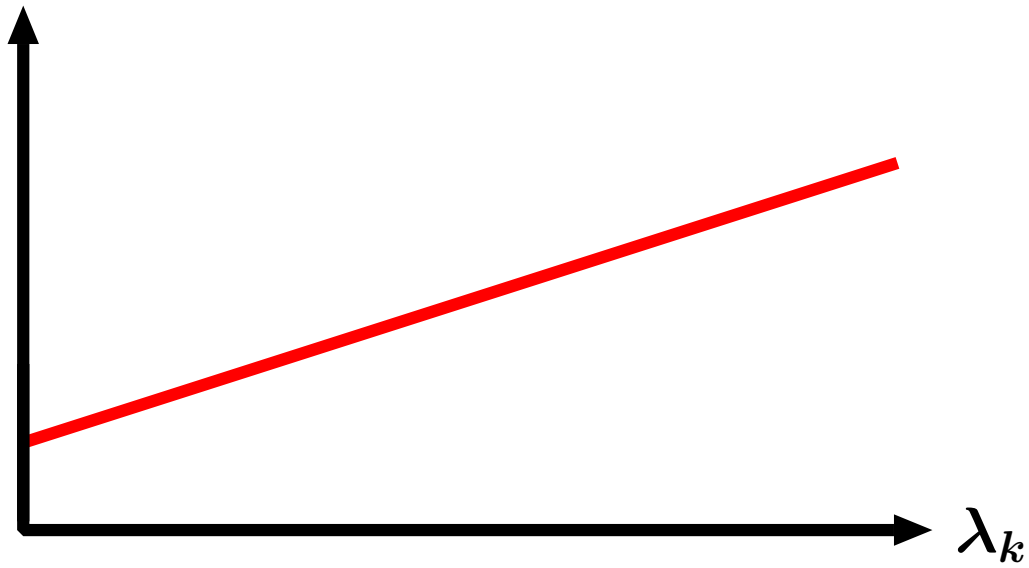


Results on newstest 08



Beispiel

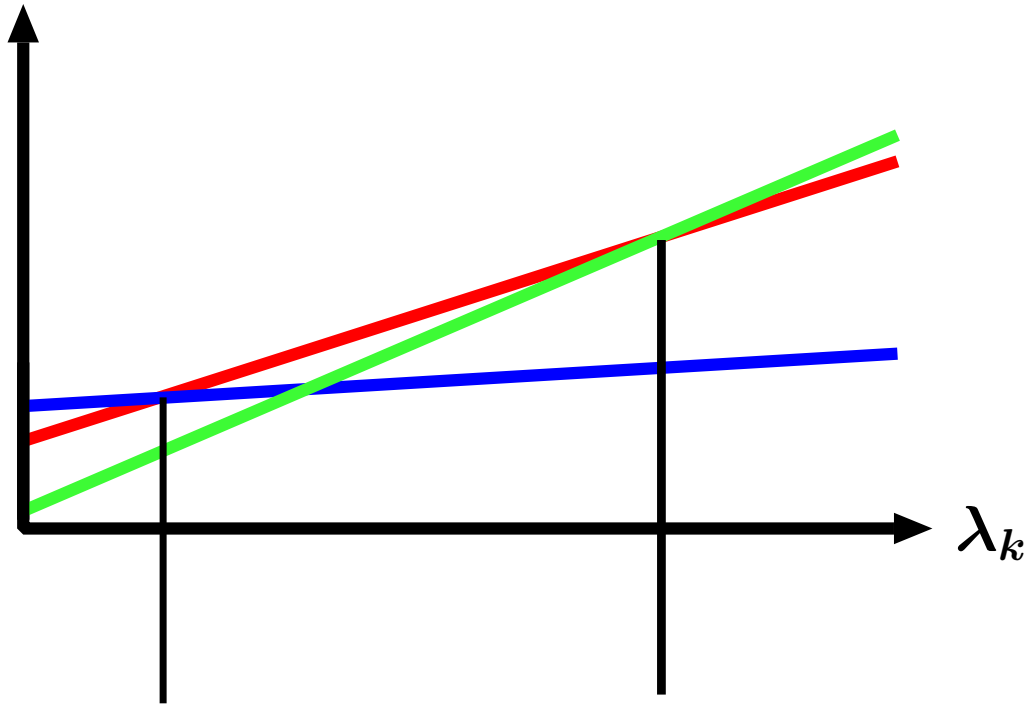
$$\sum_{m=1}^{M} \lambda_m h_m(E_1^I, F_1^J)$$



Gesamtkosten von einer einzigen Übersetzung

Beispiel

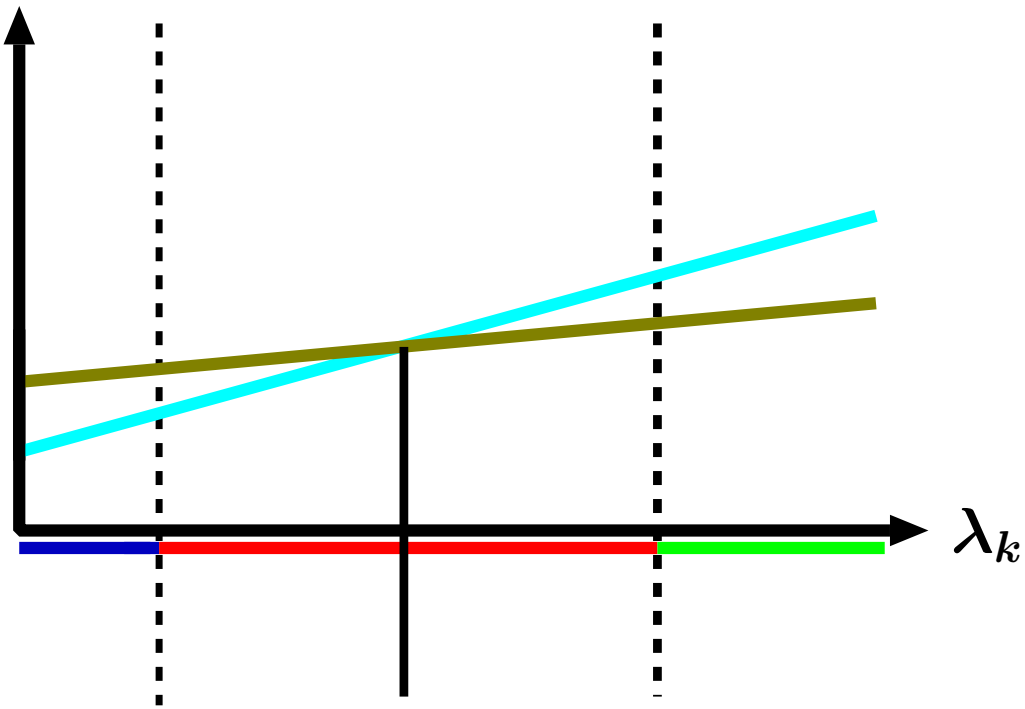
$$\sum_{m=1}^{M} \lambda_m h_m(E_1^I, F_1^J)$$



Schnittpunkte der N-Best Hypothesen für Satz Eins

Beispiel

$$\sum_{m=1}^{M} \lambda_m h_m(E_1^I, F_1^J)$$



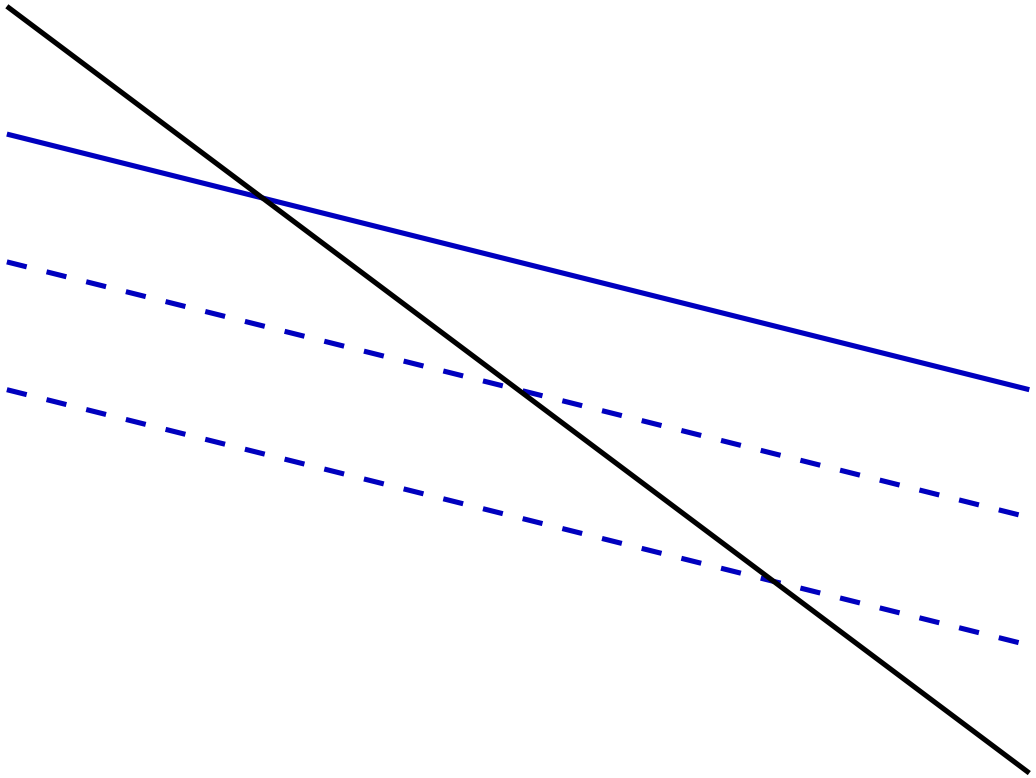
Schnittpunkte der N-Best Hypothesen für Satz Zwei

Berechnung der Konvexen Hülle

Algorithm 2: Line Sweep Algorithmus

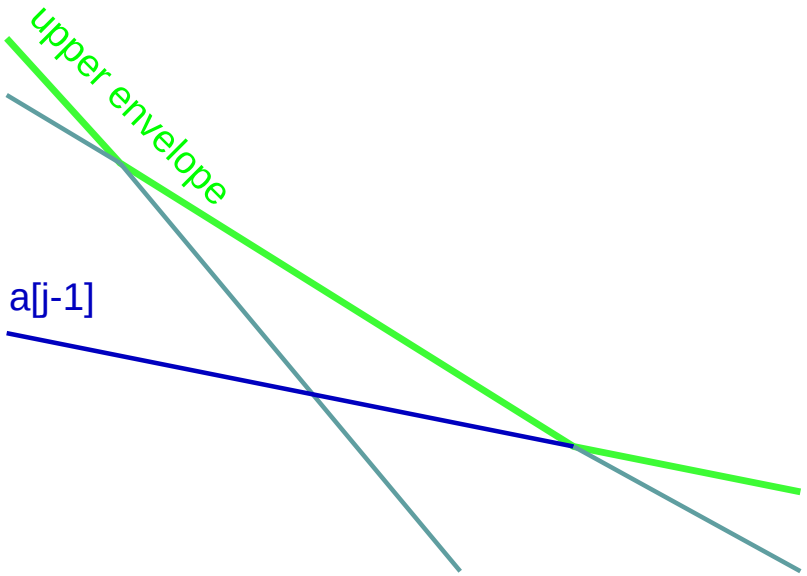
input : Array a der Größe K mit Linien sortiert nach Steigung. $j = 0$

for $(i = 0 \dots K - 1;)$ **do**
  SETZE currentIntersection = $a[i]$;
  SETZE currentIntersection.previous = minimum;
  if $0 < j$ **then**
    while *Steigung von* $a[j-1]$ *== Steigung von currentIntersection, aber Offset geringer* **do**
      VERRINGERE j;
    while $(0 < j)$ **do**
      SETZE currentIntersection.previous auf Schnittpunkt mit $a[j-1]$;
      if $a[j-1]$*.previous < currentIntersection.previous* **then**
        break;
      VERRINGERE j;
    if *0 == j* **then**
      currentIntersection.previous = minimum;
      $a[inc(j)]$ = currentIntersection;
  else
    $a[inc(j)]$ = currentIntersection;
GIB $a[0 \dots j]$ als Lösung;



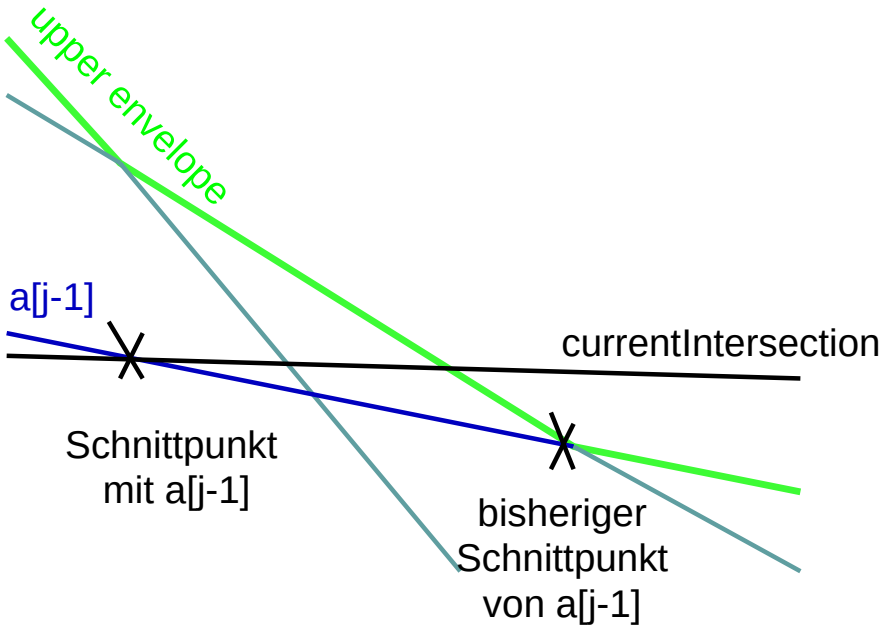
Algorithm 3: Eliminiere niedriger gelegene Linien

while *Steigung von* $a[j-1]$ *== Steigung von currentIntersection, aber Offset geringer* **do**
  VERRINGERE j;



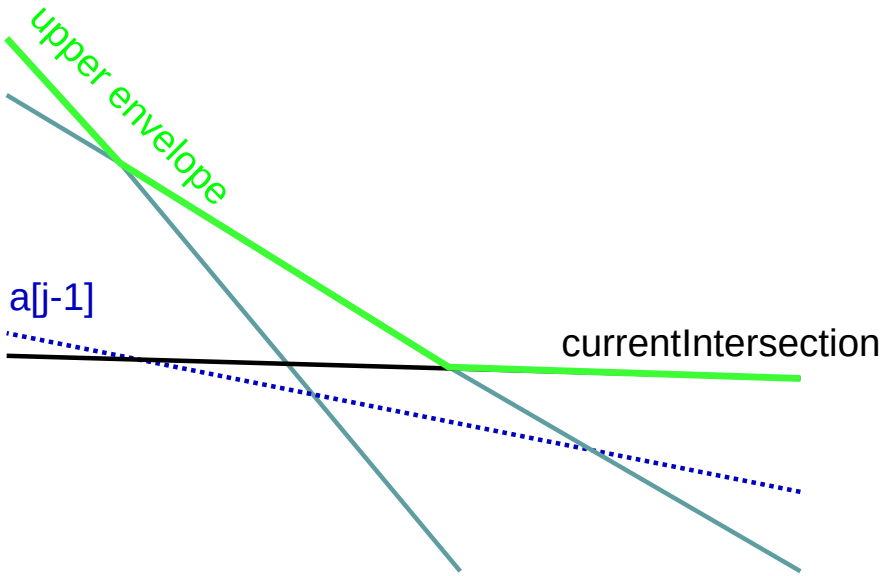
Algorithm 4: Vergleiche Schnittpunkte

while $(0 < j)$ **do**
  SETZE currentIntersection.previous auf Schnittpunkt mit $a[j-1]$;
  if $a[j-1]$.*previous* < *currentIntersection.previous* **then**
    break;
  VERRINGERE j;



Algorithm 5: Vergleiche Schnittpunkte

while $(0 < j)$ **do**
  SETZE currentIntersection.previous auf Schnittpunkt mit $a[j-1]$;
  if $a[j-1]$.*previous* < *currentIntersection.previous* **then**
    break;
  VERRINGERE j;



Algorithm 6: Vergleiche Schnittpunkte

while $(0 < j)$ **do**
  SETZE currentIntersection.previous auf Schnittpunkt mit $a[j-1]$;
  if $a[j-1]$.*previous* < *currentIntersection.previous* **then**
    break;
  VERRINGERE j;

Schnittpunkt-Berechnung

CHOR DER ÄNGSTLICHEN STUDENTEN:

Schnittpunkt-Berechnung?

[*unsicher*] Herrjemine, wie ging denn das noch?

ASSISTENTEN:

[*fürsorglich*] Der Schnittpunkt berechnet sich als

$$(\textbf{Steigung}(a) - \textbf{Steigung}(b))/(\textbf{Offset}(b) - \textbf{Offset}(a))$$

CHOR DER ÄNGSTLICHEN STUDENTEN:

[*erleichtert*] Ei der Daus, stimmt ja!

Algorithmus

Algorithm 7: MERT

```
INITIALISIERE;
while Neue Hypothese produziert wird do
    while Parameter ändern sich do
        Berechne neue (zufällige) Parameter-Reihenfolge;
        for Jeden Parameter do
            Berechne alle Schnittpunkte aller N-Best Listen;
            Berechne BLEU neu für alle Schnittpunkte (effizient!);
            Wähle besten Parameter;
    Erzeuge neue N-Best Liste mit neuen Parametern;
    Füge die neuen N-Best Einträge zu der alten Liste hinzu
```

Fragen?

Wenn ihr jetzt immer noch nicht wisst, wo unser Büro ist, haben wir was falsch gemacht :-)