Übung Algorithmische Geometrie

Lineare Programmierung

LEHRSTUHL FÜR ALGORITHMIK I · INSTITUT FÜR THEORETISCHE INFORMATIK · FAKULTÄT FÜR INFORMATIK

Andreas Gemsa

05.05.2011

Nachtrag

Nachtrag

s. Tafel

Aufgabe 1

Korrektheitsbeweis:

a) Zeige, dass jede mögliche Permutation von A gleich wahrscheinlich ist.

RandomPermutation(A)

Input: Array $A[1 \dots n]$
Output: Array A, zufällig gleichverteilt permutiert
for $k \leftarrow n$ **to** 2 **do**
  $r \leftarrow$ Random(k)
  tausche $A[r]$ und $A[k]$

Aufgabe 1

Korrektheitsbeweis:

b) Zeige, dass die Aussage aus a) nicht stimmt, wenn wir in der 2. Zeile k mit n ersetzen.

RandomPermutation(A)

Input: Array $A[1 \ldots n]$
Output: Array A, zufällig gleichverteilt permutiert
for $k \leftarrow n$ **to** 2 **do**
    $r \leftarrow$ Random(k)
    tausche $A[r]$ und $A[k]$

Aufgabe 1

Korrektheitsbeweis:

b) Zeige, dass die Aussage aus a) nicht stimmt, wenn wir in der 2. Zeile k mit n ersetzen.

RandomPermutation(A)

Input: Array $A[1 \ldots n]$
Output: Array A, zufällig gleichverteilt permutiert
for $k \leftarrow n$ **to** 2 **do**
  $r \leftarrow$ Random(n)
  tausche $A[r]$ und $A[k]$

Aufgabe 2

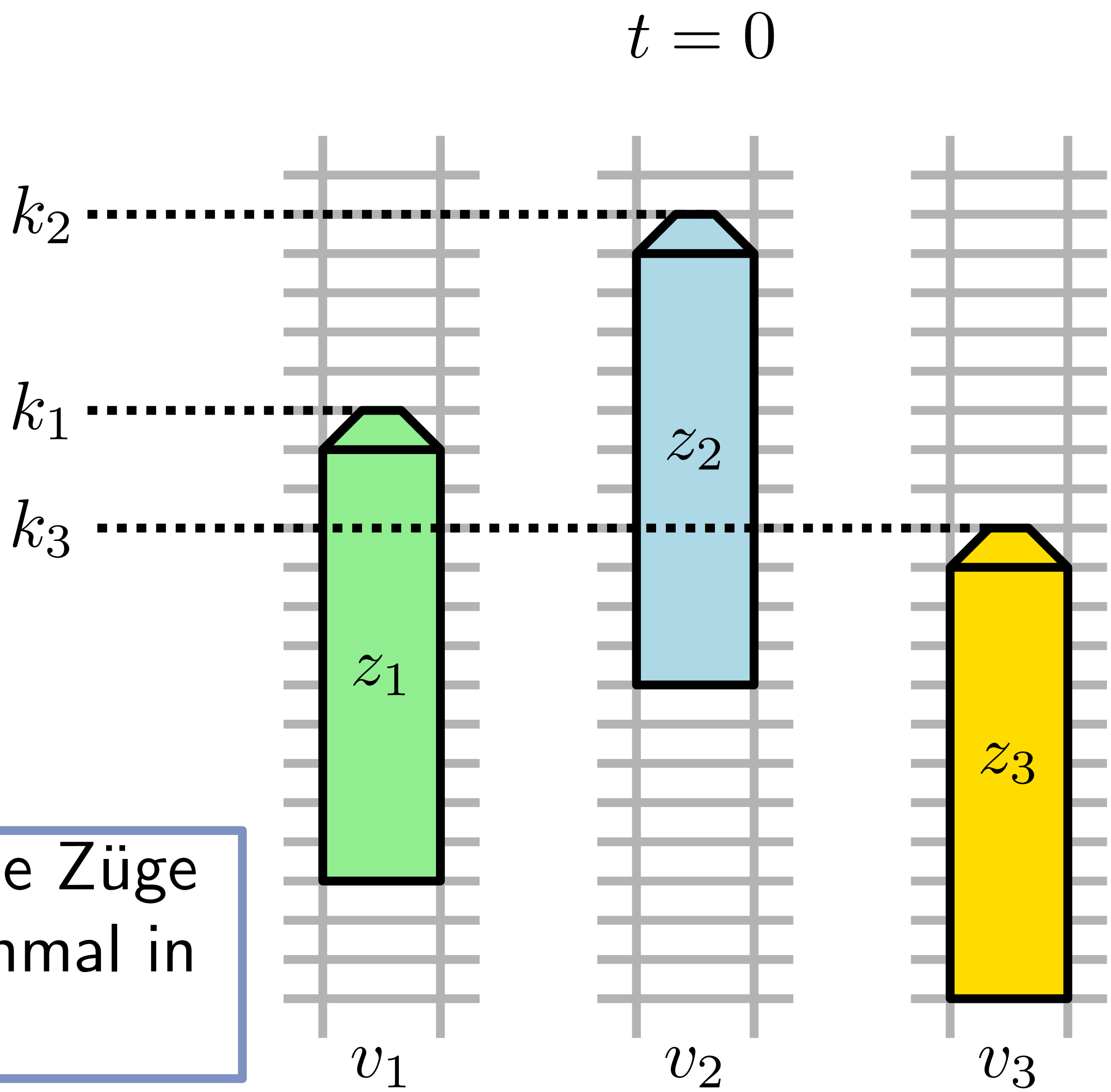
ParanoidMax

Input: Endliche Menge $A \subset \mathbb{R}$
Output: Maximum $\max_{a \in A} a$ der Menge
if $|A| = 1$ **then**
    return einziges Element $a \in A$
else
    $a =$ zufällig gewähltes Element aus A
    $b =$ ParanoidMax$(A \setminus \{a\})$
    if $b \geq a$ **then**
        return b
    else
        prüfe unnötigerweise jedes Element aus $A \setminus \{a\}$, um sicherzugehen, dass a wirklich größer ist
        return a

a) Asymptotische w-c Laufzeit?

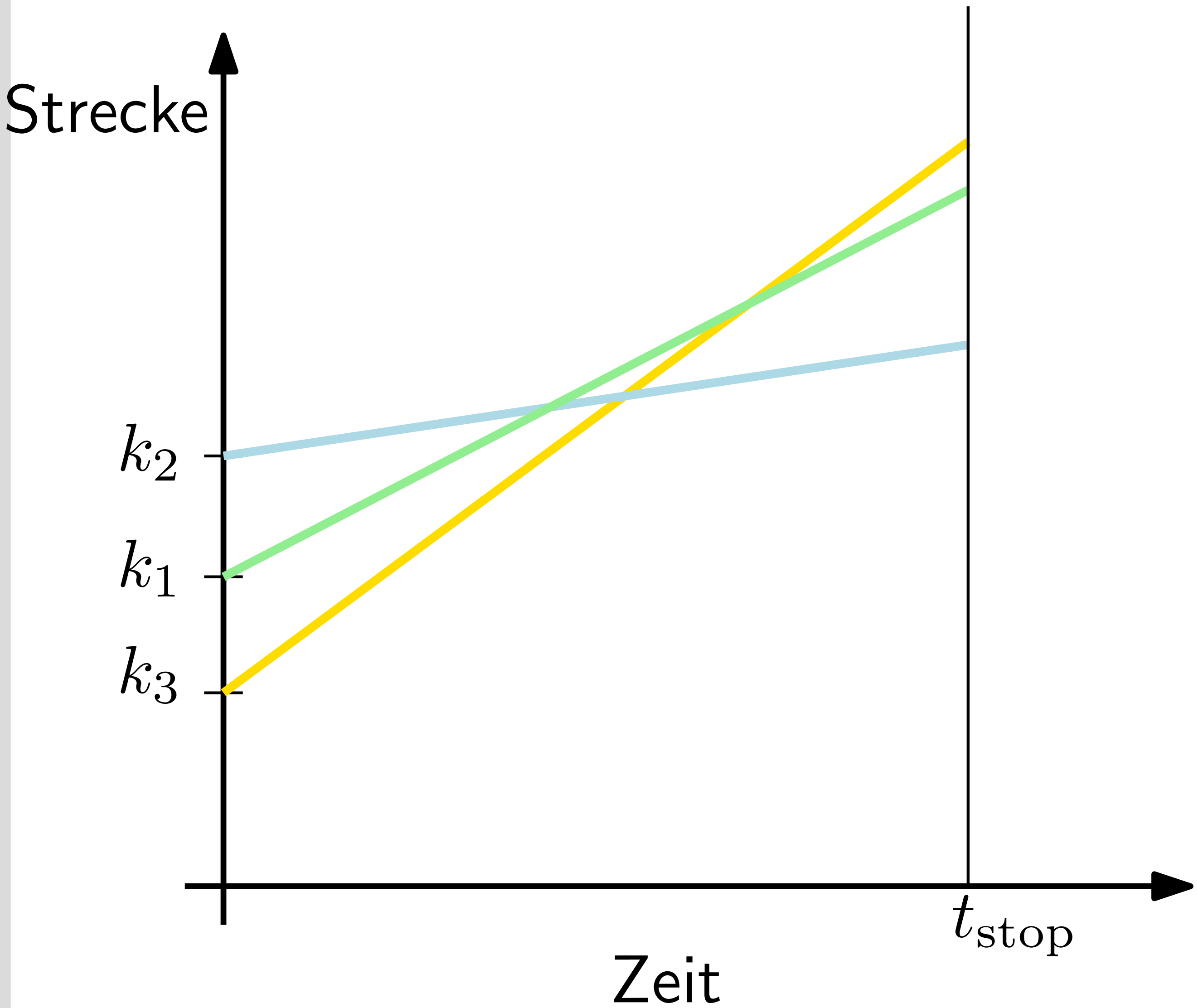
b) erwartete Laufzeit echt besser als w-c

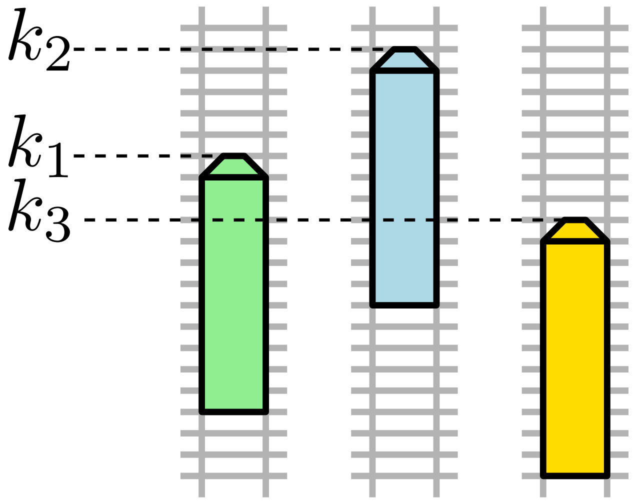
Züge



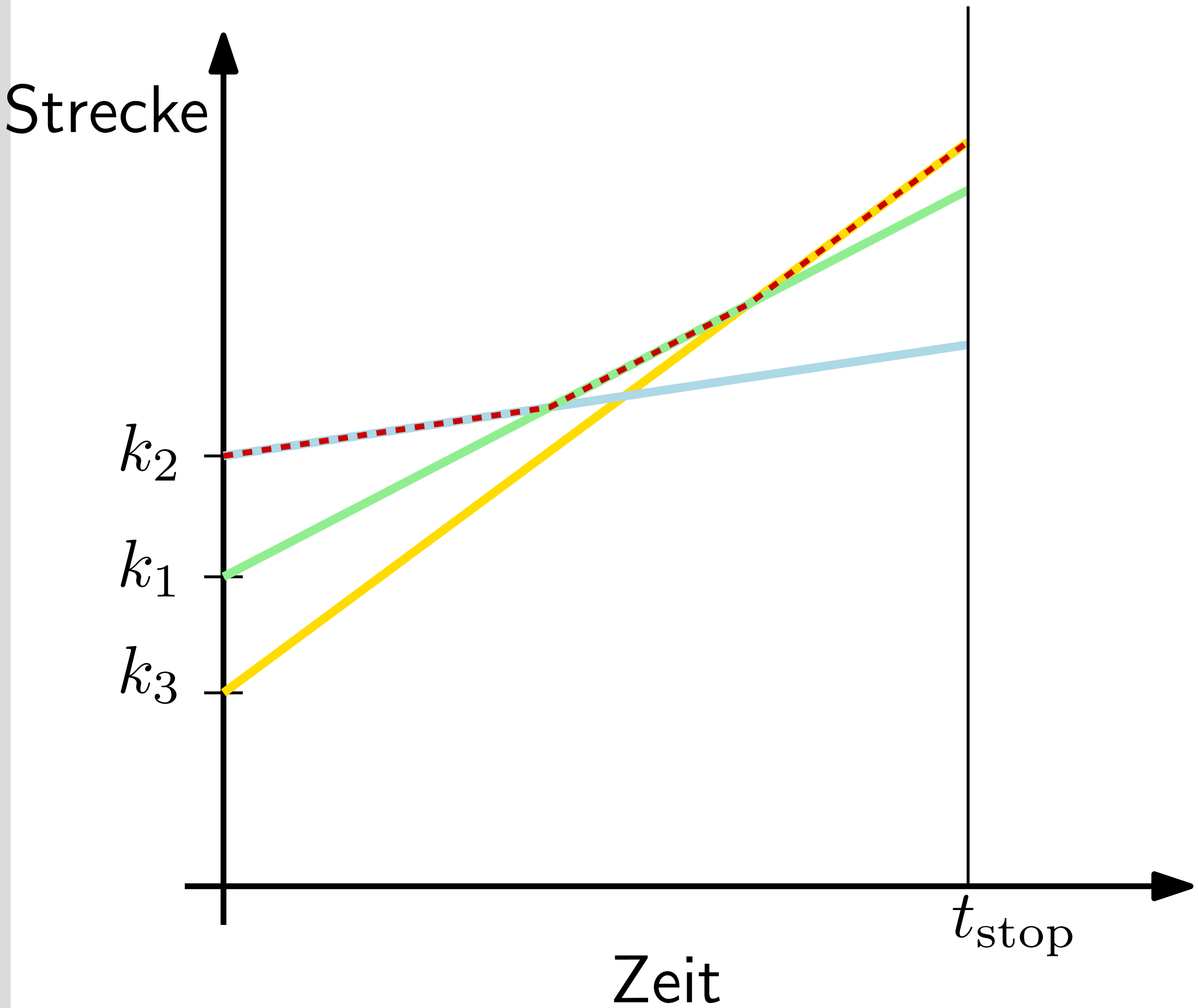
Algorithmus: welche Züge bis Zeitpunkt t_s einmal in Führung

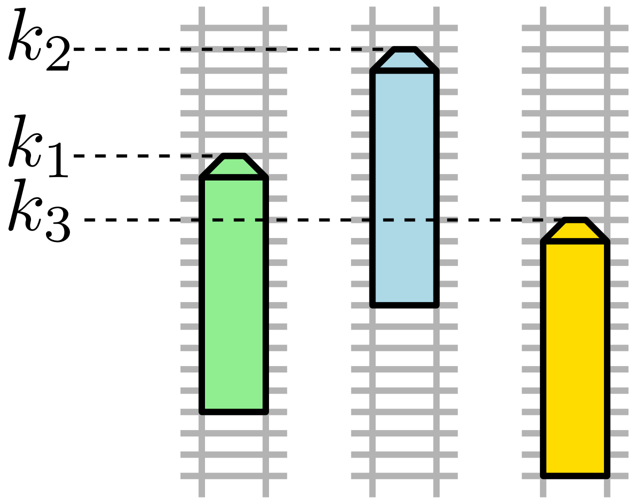
Züge



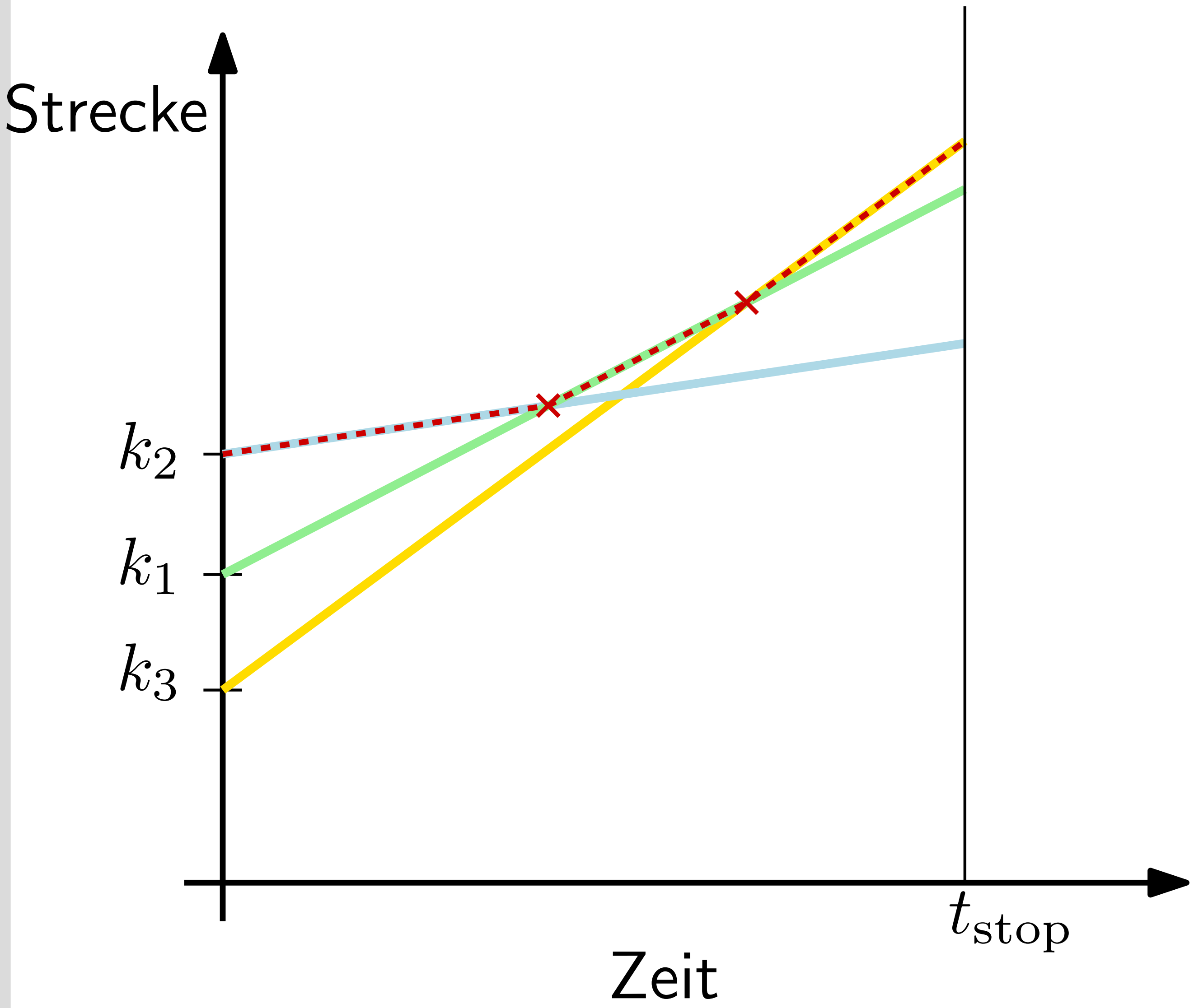


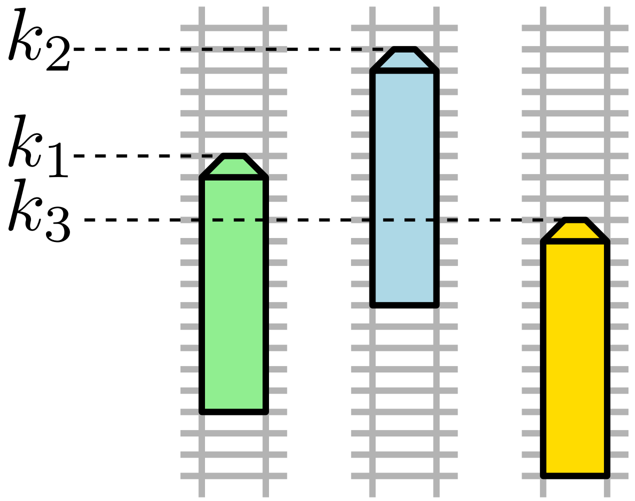
Züge



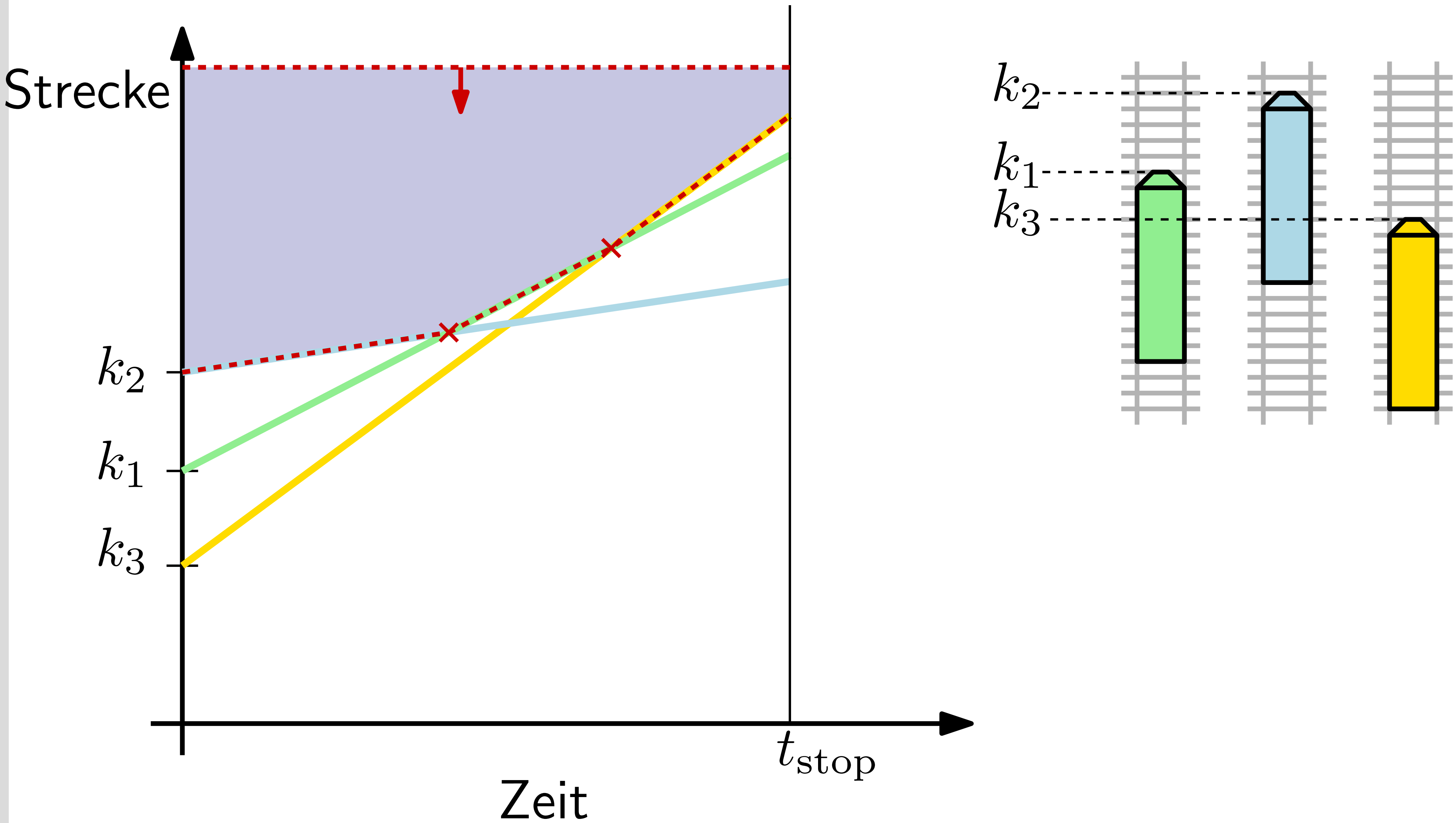


Züge

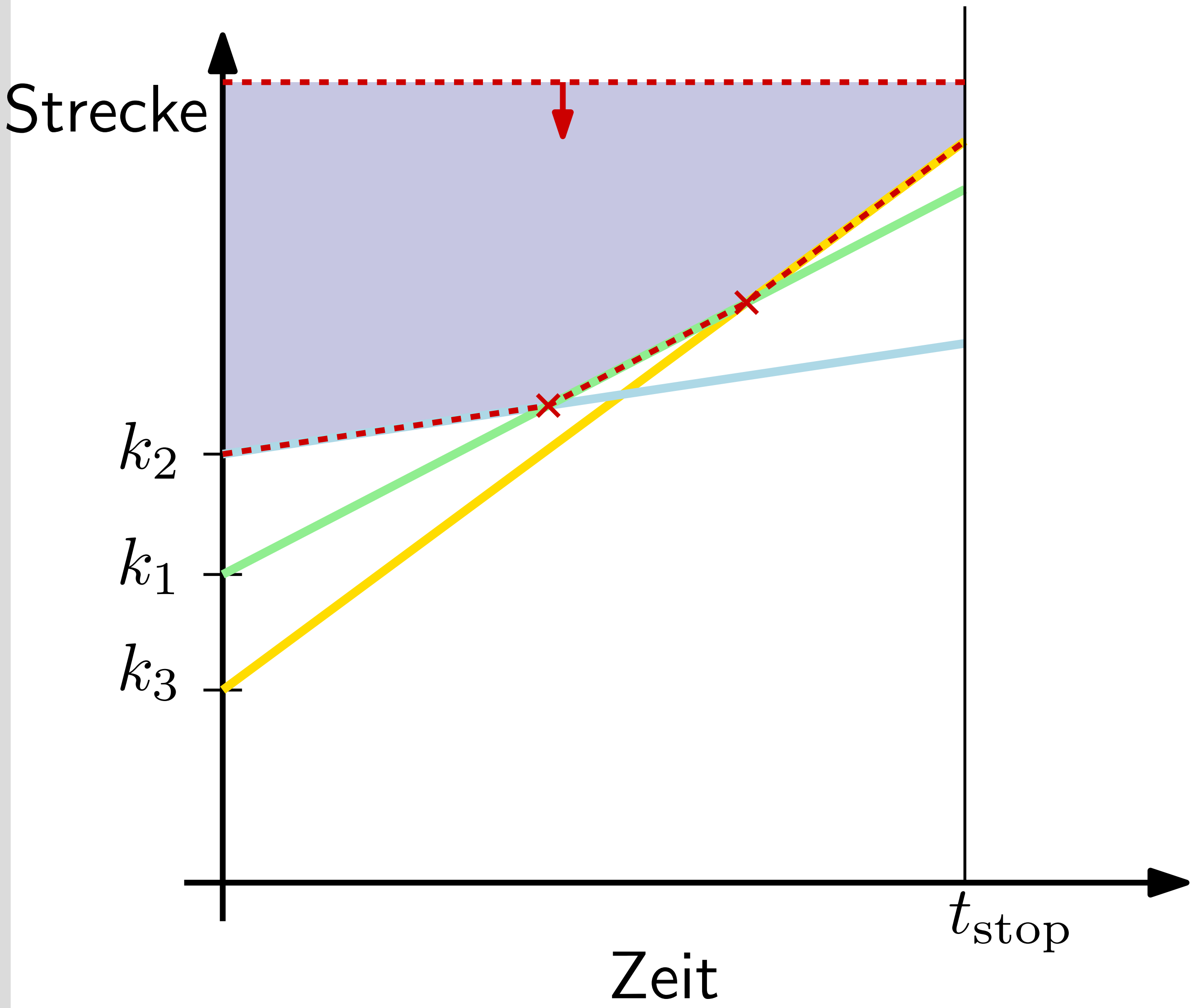


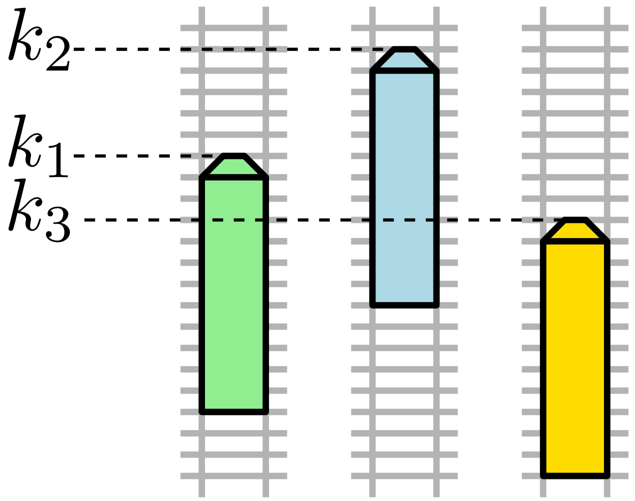


Züge



Züge





Züge

IntersectHalfplanes(H)

if $|H| = 1$ **then**
  $C \leftarrow H$
else
  $(H_1, H_2) \leftarrow$ SplitInHalves(H)
  $C_1 \leftarrow$ IntersectHalfplanes(H_1)
  $C_2 \leftarrow$ IntersectHalfplanes(H_2)
  $C \leftarrow$ IntersectConvexRegions(C_1, C_2)
return C

Das war's!

Fragen?

Nächster Termin:

Donnertag, 19.05, 10:15 Uhr

Raum 131, Gebäude 50.34