Übung 4 – 31.5.2017

Branch-and-Bound
- ↳ Beispiel ~~Gurke~~ Knapsack
- ↳ Euklidischer TSP
  - ↳ untere Schranken
  - ↳ obere Schranken
  - ↳ BnB Algo
  - ↳ zulässige (Teil-)Lösungen?

Branch and Bound Knapsack

Formaler Algorithmus morgen in der Vorlesung.

Informell für ein Item ℓ

1. Prüfe Zulässigkeit der ersten $\ell-1$ Items.
2. Lösung verbessert? P ≙ beste Lösung bisher
3. Return, falls im Blatt angekommen
4. Berechne UB = obere Schranke → z.B. durch Fractional Greedy! (Alg 1.4)
   optional: (LB = untere Schranke) → z.B. durch Greedy.
   ↳ P = LB, falls P < LB
5. Falls UB ≤ P return

   ansonsten
   a) Prüfe Lösung mit $b_\ell = 0$
   b) und mit $b_\ell = 1$
6. Return

Verbesserung zu früher

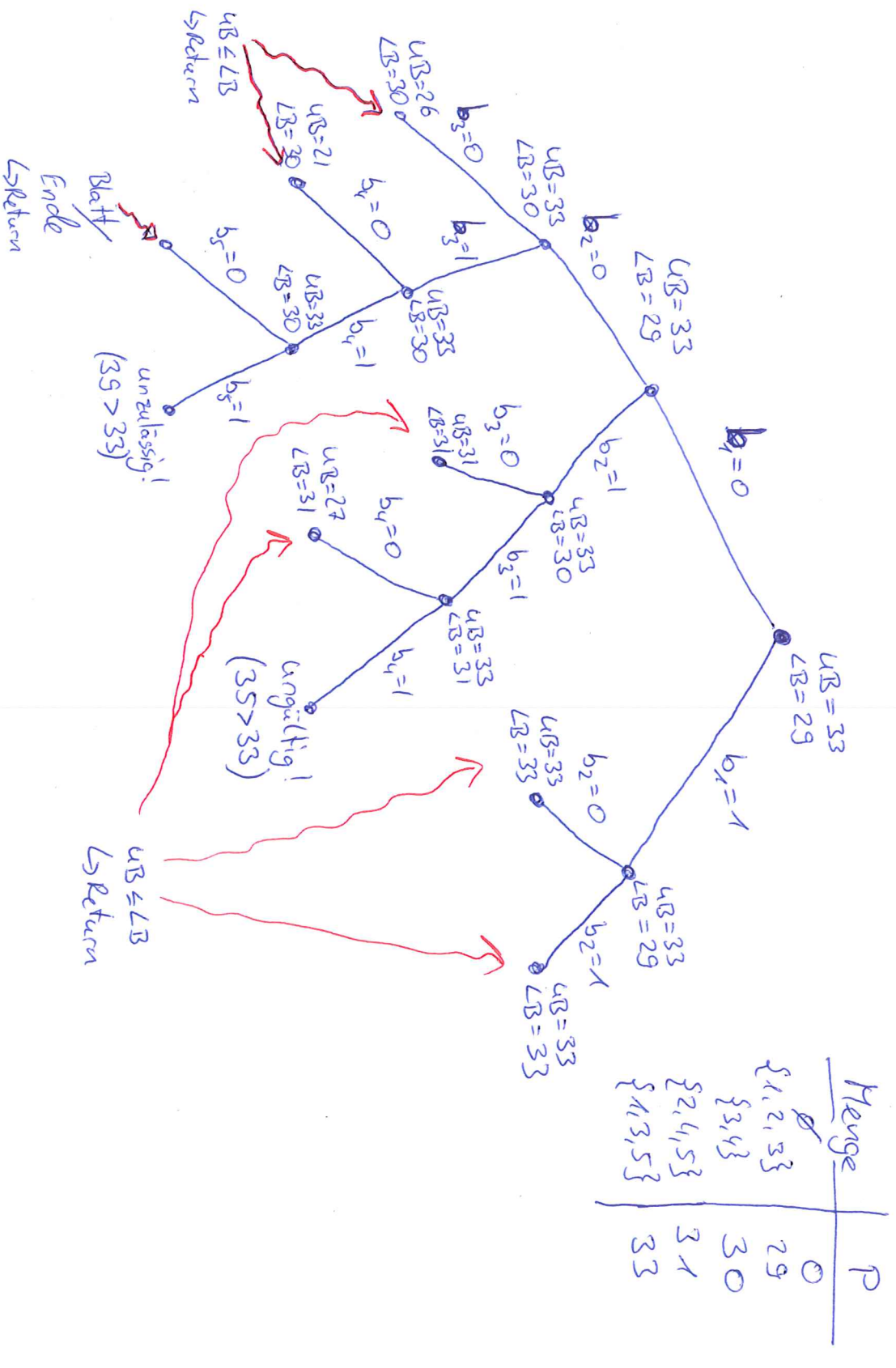
Nimm solange Items rein solange es geht, d.h. betrachte jedes Item einmal!

später: Greedy 0

Knapsack Instanz:

i	1	2	3	4	5
z_i	11	5	13	17	9
p_i	11	5	13	17	9

$z = 33$



Menge	P
$\emptyset$	0
{1,2,3}	29
{3,4}	30
{2,4,5}	31
{1,3,5}	33

Branch and Bound Algorithmen sind immer ähnlich aufgebaut:

1. Zulässigkeit testen
2. Test auf bessere Lösung
3. Schranken betrachten
4. Verzweigen falls nötig.

Herausforderung: Gute Schranken finden!
D.h. Differenz aus oberer und unterer Schranke so klein wie möglich halten!

Euklidisches Traveling Salesman Problem (ETSP)

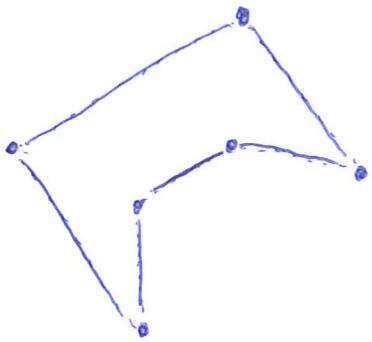
Betrachte folgendes Problem:

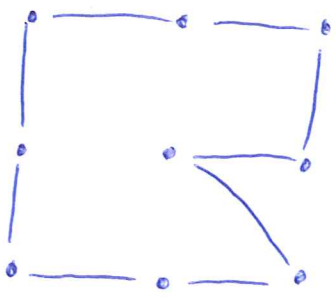
Gegeben: n Punkte $P_1, \dots, P_n$ in der Ebene, d.h. $P_i \in \mathbb{R}^2$

Gesucht: Eine kürzeste (→ also minimieren!) Rundreise, die alle Punkte abläuft

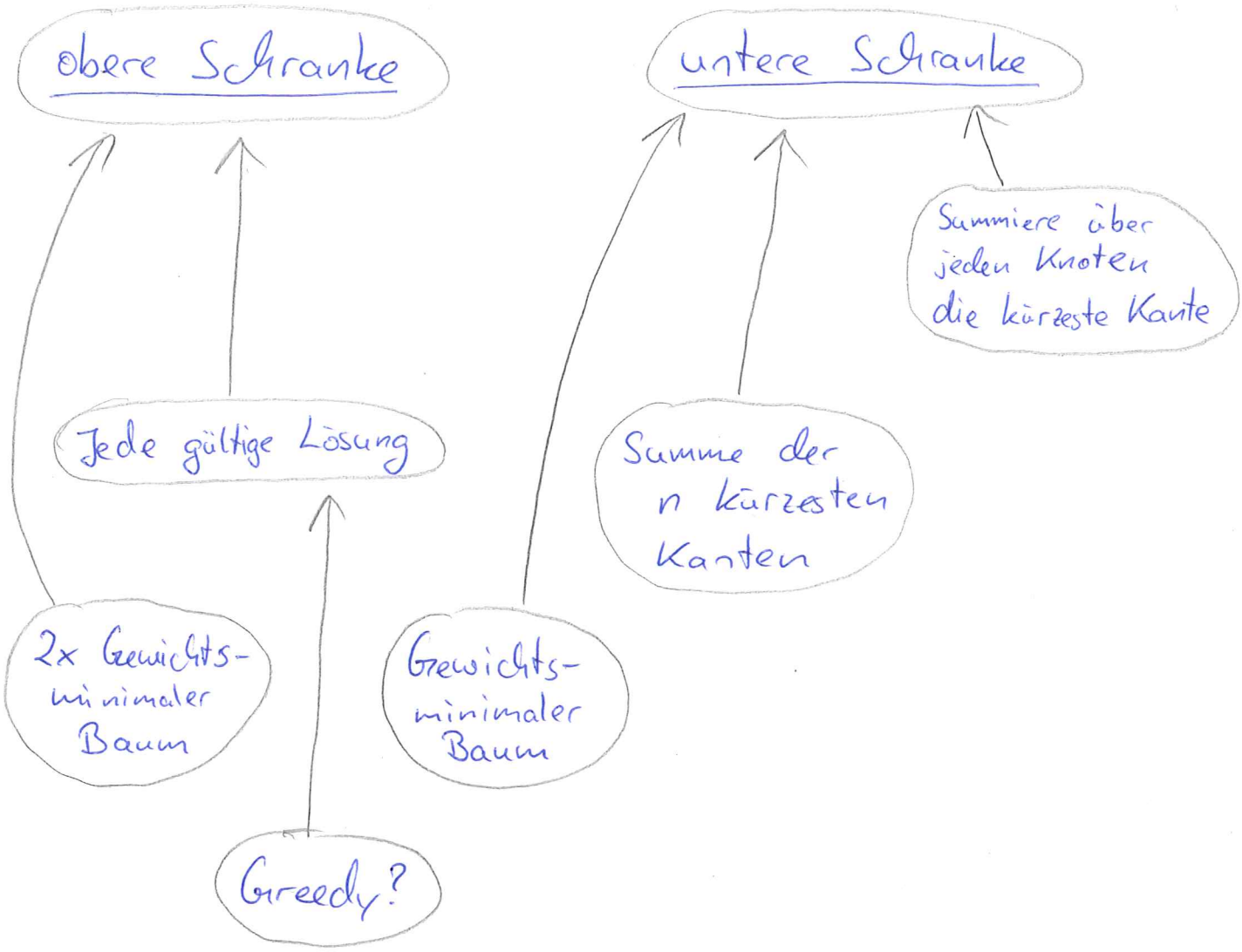
(kürzeste: euklidische Distanz.)

Beispiele:





Was sind mögliche Schranken?



Wie sieht ein möglicher Greedy aus?

Sei P die Menge der Punkte.

Starte bei p_1 und entferne p_1 aus P

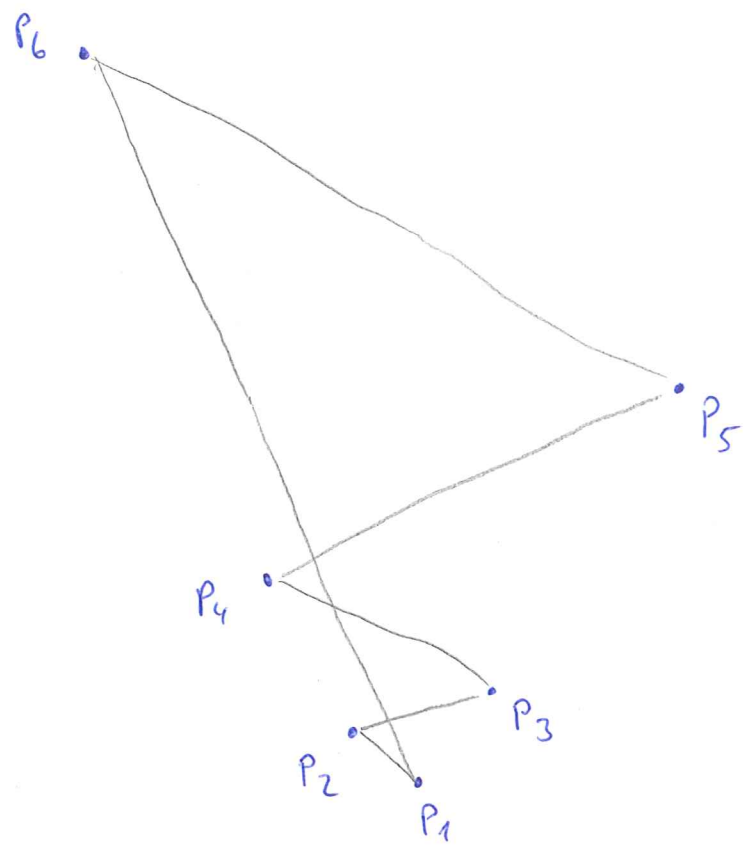
Setze a auf p_1

Solange P nicht leer ($P \neq \emptyset$)

- sei $p_j \in P$ der am nächsten an a liegende Punkt
- Füge Kante $\{a, p_j\}$ hinzu
- entferne p_j aus P
- setze a auf p_j

Füge Kante $\{a, p_1\}$ hinzu.

Beispiel für Greedy:



Nun, der BnB-Algorithmus:

Worüber verzweigen wir?

Oder anders: Was ist entweder da (1) oder nicht da (0)?

Antwort: Die Kanten!

Sei $b_{i,j} \in \{0,1\}$ für Kante von p_i nach p_j mit $i \neq j$.

Jetzt können wir mit einer beliebigen Reihenfolge der Kanten den BnB aufschreiben:

1. ~~Ist die Teillösung zulässig?~~ → ~~wie testen?~~
2. Gibt es eine ~~bessere~~ Lösung?
3. ~~Schranken~~ betrachten
4. Verzweigen!

} Das können wir!

~~Vollständiger Algo:~~

~~Rekursiv~~

übergeben:

~~n~~

n Punkte $P_1, \dots, P_n$

K

l

~~$x_1 = b_1$ ... $x_l = b_l$~~

$x_j = b_j$ für ersten $l-1$ Kanten
mit $b_j \in \{0, 1\}$

Ausgabe: Besten Lösungswert K

ETSP(l):

1. If not (valid(l)) then return
2. Berechne $U := UB(b_1, \dots, b_{l-1})$
   If $(U < K)$ then $K := U$
3. Berechne $L := LB(b_1, \dots, b_{l-1})$
4. If $(l > \binom{n}{2})$ then return
5. If (~~K~~ $L < K$) then
   $b_l = 0$
   ETSP($l+1$)
   $b_l = 1$
   ETSP($l+1$)
6. Return!

$\binom{n}{2} = \frac{n(n-1)}{2}$ $\hat{=}$ Anzahl möglicher Kanten.

Test auf Zulässigkeit:

valid(ℓ){

1. if $\left(\sum_{i=1}^{\ell-1} b_i > n\right)$ return false // mehr als n Kanten ausgewählt.

2. for $i=1$ to n do

   if $\left(\sum_{j=1}^{\ell-1} (j\text{-te Kante liegt an Knoten } p_i) \cdot b_j > 2\right)$ then

   Grad des Knotens/Punktes

   return false

   Der Grad darf maximal 2 sein!

3. if $\left(\sum_{i=1}^{\ell-1} b_i < n\right)$ then

   if (Lösung kreisfrei) then

   return true

   else

   return false

   } Bei $< n$ Kanten muss Lösung kreisfrei sein. Test z.B. über Breitensuche aus AuD I

4. if (Genau eine Zusammenhangskomponente) then

   return true

   else

   return false

   } n Kanten ausgewählt