Agenda

1. Einführung und Übersicht

2. Lineare Optimierung

3. Graphentheorie

4. Ganzzahlige Optimierung

5. Dynamische Optimierung

Auf dem Tisch liegen 18 Streichhölzer. Die Spielregeln: Die erste Spielerin nimmt entweder ein, zwei oder drei Hölzer vom Tisch, danach nimmt der zweite Spieler entweder ein, zwei oder drei der verbliebenen Hölzer und so weiter. Verloren hat diejenige Person, die das letzte Streichholz vom Tisch nimmt.

Wie gewinne ich das Spiel, wenn ich beginnen darf?

Gewinnstrategien für das Streichholzspiel

i	1	2	3	4
$GS[i]$	N	J	J	J

Algorithm: Berechnung der Gewinnstrategie

function Gewinnstrategie(X)

$GS[1] := N, GS[2] := J, GS[3] := J$

$i := 3$

while $i < X$ **do**

i:=i+1

if $GS[i-3] = GS[i-2] = GS[i-1] = J$ **then**

$GS[i] := N$

else

$GS[i] := J$

i	1	2	3	4
$GS[i]$	N	J	J	J

Algorithm: Berechnung der Gewinnstrategie

function Gewinnstrategie(X)

$GS[1] := N, GS[2] := J, GS[3] := J$

$i := 3$

while $i < X$ **do**

  i:=i+1

  if $GS[i-3] = GS[i-2] = GS[i-1] = J$ **then**

    $GS[i] := N$

  else

    $GS[i] := J$

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$GS[i]$	N	J	J	J	N	J	J	J	N	J	J	J	N	J	J	J	N	J

动态优化
定义：最优性原理
“一个最优策略具有这样的性质，即无论初始状态和初始决策是什么，剩余的决策都必须构成相对于第一个决策所产生状态的最优策略。”
Richard Bellman (1957): 动态规划
• 动态优化是一个通用原则。

Definition: Principle of Optimality

„An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.“

Richard Bellman (1957): Dynamic Programming

• 要应用动态优化，必须对问题进行适当的建模。
• 优化模型被分为阶段（周期，阶段）。
• 在每个阶段，存在一系列状态，分别进行考虑。
• 基于这些观察，做出决策。
• 决策的结果：在下一个阶段，系统处于另一个状态。
• 决策的总和称为一种策略。
• 因此，目标函数受整个策略的影响，而不仅仅是部分决策的

- Bei der dynamischen Optimierung handelt es sich um ein allgemeines Prinzip.
- Um die dynamische Optimierung anwenden zu können, muss eine geeignete Modellierung des Problems vorliegen.
- Das Optimierungsmodell ist in Stufen (Perioden, Etappen) eingeteilt.
- Auf jeder Stufe existiert eine Anzahl von Zuständen, die separat betrachtet werden.
- Auf Grund dieser Beobachtungen wird eine Entscheidung getroffen.
- Folge der Entscheidung: Auf der nächsten Stufe befindet sich das System in einem anderen Zustand.
- Die Summe der Entscheidungen nennt man eine *Politik*.
- Die Zielfunktion wird also von der gesamten Politik beeinflusst, nicht nur von einem Teil der Entscheidungen.

Grundprinzip:

- Die Optimalität der Entscheidung auf einer Stufe hängt nicht von den vorhergehenden Entscheidungen ab.
- Eine optimale Politik bedeutet: Die Folge der Entscheidungen ab einer Stufe bis zur letzten Stufe ist optimal bezüglich des auf dieser Stufe beobachteten Zustandes.

Idee: Optimalitätsprinzip von Bellman

- Entscheidungen auf der k-ten Stufe sind abhängig vom Optimalwert der Stufe k+1.
- Die Optimierung beginnt also auf der letzten Stufe und setzt sich dann rückwärts fort (Rückwärtsrechnung).
- Sind alle Stufen abgearbeitet sucht man nun ausgehend von der ersten Stufe die optimale Politik (Vorwärtsrechnung).

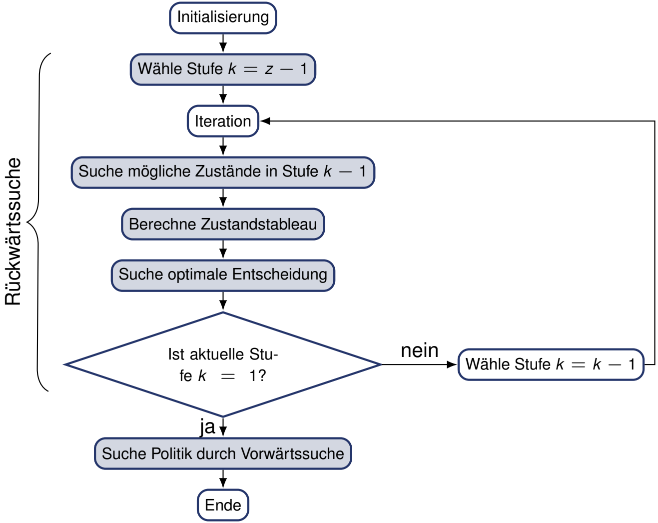
基本原则:

- 在某一阶段的决策的最优性不取决于之前的决策。
- 最优策略意味着：从某一阶段到最后一阶段的决策序列相对于该阶段的观察状态是最优的。

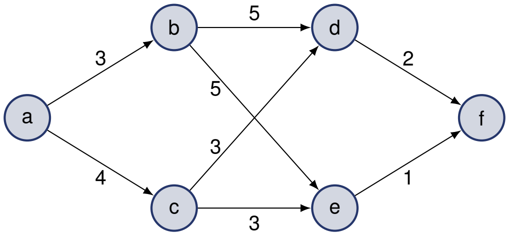
理念：贝尔曼的最优性原则

- 第k阶段的决策取决于第k+1阶段的最优值。
- 优化从最后一阶段开始，然后向后进行（反向计算）。
- 当所有阶段都处理完毕后，现在从第一阶段开始寻找最优策略（前向计算）。

Dynamische Optimierung mit n Stufen

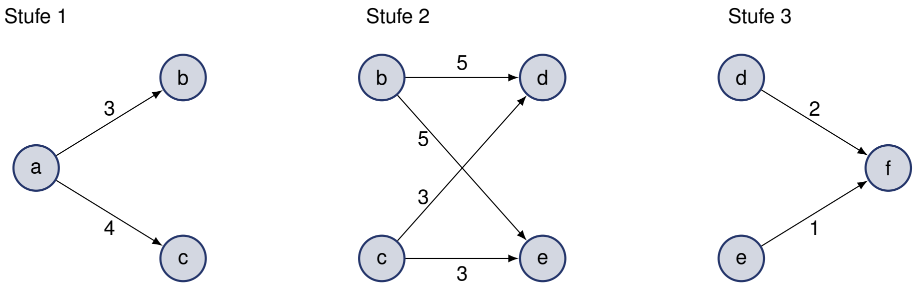


Finde den kürzesten Weg von Knoten *a* nach *f* mit Hilfe der dynamischen Optimierung:



Stufe 1 Stufe 2 Stufe 3 Stufe 4

Dynamische Optimierung – Beispiel

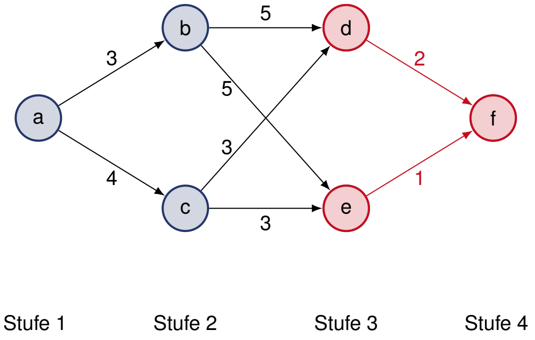


Dynamische Optimierung mit $z = 4$ Stufen

Dynamische Optimierung – Beispiel

$k = 3$

	f	x_3^*	$c_3^*(x)$
d	2	f	2
e	1	f	1

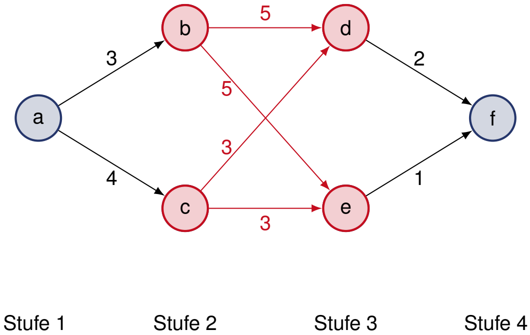


$k = 3$

	f	x_3^*	$c_3^*(x)$
d	2	f	2
e	1	f	1

$k = 2$

	d	e	x_2^*	$c_2^*(x)$
b	7	6	e	6
c	5	4	e	4



$k = 3$

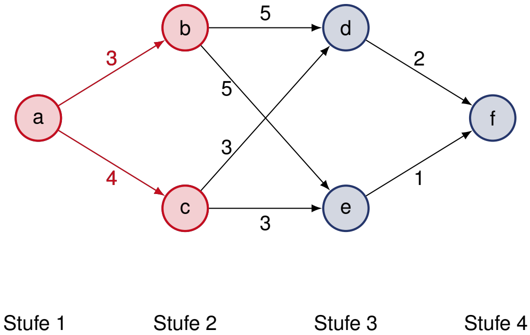
	f	x_3^*	$c_3^*(x)$
d	2	f	2
e	1	f	1

$k = 2$

	d	e	x_2^*	$c_2^*(x)$
b	7	6	e	6
c	5	4	e	4

$k = 1$

	b	c	x_1^*	$c_1^*(x)$
a	9	8	c	8



Dynamische Optimierung – Beispiel

$k = 3$

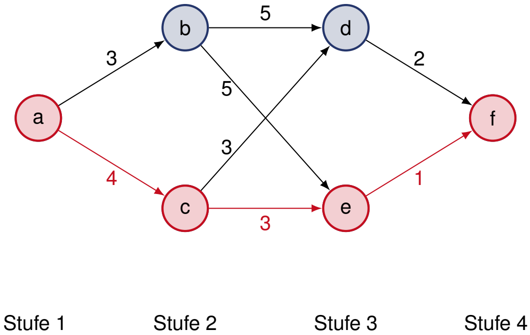
	f	x_3^*	$c_3^*(x)$
d	2	f	2
e	1	f	1

$k = 2$

	d	e	x_2^*	$c_2^*(x)$
b	7	6	e	6
c	5	4	e	4

$k = 1$

	b	c	x_1^*	$c_1^*(x)$
a	9	8	c	8



Dynamische Optimierung

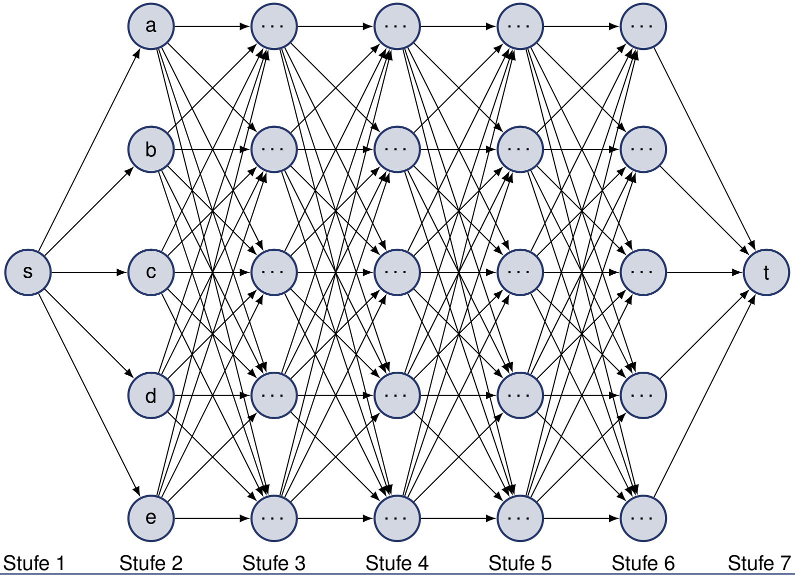
Ähnlich der bereits behandelten Suche nach dem kürzesten Weg führt die dynamische Optimierung schneller ans Ziel als die vollständige Enumeration.

Vorteile bieten sich erst bei der Behandlung von nicht trivialen Netzwerken.

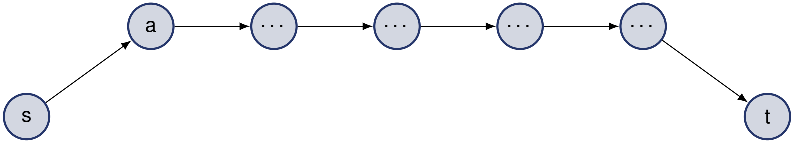
Schon bei einem Netzwerk mit 27 Knoten auf sieben Stufen ist der Unterschied beachtlich.

类似于已经讨论过的寻找最短路径，动态优化比完全枚举更快地达到目标。
优势在于处理非平凡网络时才显现。
即使在一个具有27个节点和七个阶段的网络中，差异也是显著的。

Dynamische Optimierung

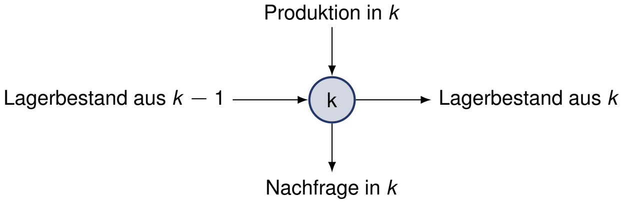


- Gesucht ist der kürzeste Weg von Knoten s zu Knoten t.
- Es existieren insgesamt $5^5 = 3.125$ verschiedene Pfade.
- Für die Bestimmung einer Weglänge müssen allerdings nur 5 Additionen durchgeführt werden.
  - Für die dynamische Optimierung werden also 105 Additionen durchgeführt.
  - Die vollständige Enumeration wäre also insgesamt 3.125/105 = 30-fach aufwändiger.



Lagerhaltungsproblem

- Mehrstufige Optimierung
- Ziel: Minimierung der Produktions- und Lagerkosten
- Befriedigung der Nachfrage in jeder Periode k aus Produktion oder Lager
- Festlegung der Produktionsmenge in jeder Periode k
- Mengenbeschränkungen
  - Produktion
  - Lager
- Kosten
  - Produktion
    - Fixkosten
    - Stückkosten
  - Lager
    - Stückkosten pro Periode



Lagerhaltungsproblem – Beispiel

Mithilfe eines ERP kann die Nachfrage aller Kund*innen einer Medizintechnikherstellerin für die nächste Produktionsperiode – vier Monate – prognostiziert werden. Für ein bestimmtes Gerät wurde eine Prognose für die Nachfrage d ermittelt.

Ziel: Minimierung der Produktions- und Lagerkosten bei Befriedigung der Nachfrage d.

Mengenbeschränkungen pro Monat k

- Produktion x: 5 Einheiten
- Lager i: 4 Einheiten

Kosten

- Produktion $c(x)$
  - Fixkosten pro Periode, in der produziert wird: 3.000 Euro
  - Stückkosten: 1.000 Euro
- Lager $c(i)$
  - Stückkosten pro Periode: 500 Euro

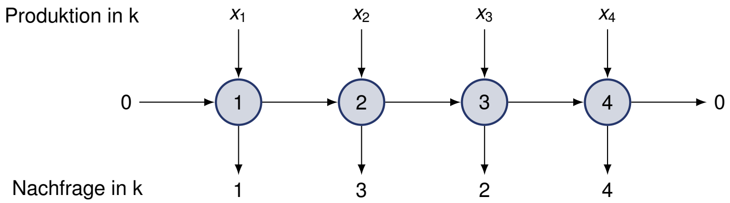
Monat	Nachfrage d
1	1
2	3
3	2
4	4

使用企业资源规划（ERP）系统，可以预测医疗技术制造商下一个生产周期（四个月）所有客户的需求。对于特定设备，已确定了下一个生产周期的需求预测。
目标：在满足需求d的情况下，最小化生产和库存成本。每月的生产量限制k

- 生产量x：5个单位
- 库存i：4个单位成本
- 生产成本c(x)
  - 每个生产周期的固定成本：3000欧元
  - 单位成本：1000欧元
- 库存成本c(i)
  - 每周期单位成本：500欧元

月份
需求d

Lagerhaltungsproblem – Beispiel



- Stufen: Perioden
- Zustände: Lagerbestand zu Beginn der Periode
- Anwendung der dynamischen Programmierung zur Lösung des Problems
- Die einzelnen Produktionsperioden werden hier als Stufen angesehen, die Zustände in den einzelnen Perioden entsprechen dem in der jeweils vorherigen Periode festgelegten Lagerbestand.

- Stufen：周期
- Zustände：周期开始时的库存量
- 应用动态规划解决问题
- 这里将每个生产周期视为阶段，每个阶段的状态对应于前一个周期中确定的库存量。

Lagerhaltungsproblem – Beispiel

Zunächst Betrachtung des vierten Monats

- Letzte Periode, daher keine Lagerhaltung zum Ende der Periode
- Befriedigung der Nachfrage von 4 Einheiten
- Produktionskosten entsprechend

- 首先考虑第四个月
- 最后一个周期，因此在周期结束时没有库存
- 满足 4 个单位的需求
- 对应的生产成本

$k = 4$

	0	x_4^*	$c_4^*(x)$
0	7	0	7
1	6	0	6
2	5	0	5
3	4	0	4
4	0	0	0

固定成本3k和1k单个成本

- 这个表格代表第四个月的决策。
- 第一列显示了初始库存i的不同可能值（从0到4单位）。
- 第二列 x_4^* 显示了在这个月中对于每个初始库存水平来说，最优的生产决策（即生产多少单位）。
- 第三列显示了对应的总成本 $c_4^*(x)$，这个成本包括生产成本和库存成本。
- 因为这是最后一个时间段，所以没有未来的库存成本需要考虑。

Lagerhaltungsproblem – Beispiel

- 这个表格代表第三个月的决策。
- 第一列显示了初始库存i的不同可能值。
- 第二列和之后的列显示了不同的生产量x选择，以及对应的成本计算。每个格子中的计算考虑了生产成本 $c(x)$ 和未来库存成本 $f_{t+1}(i+x-d)$，其中 f_{t+1} 是未来的最小成本函数。

Rückschritt zum dritten Monat

- Befriedigung der Nachfrage von 2 Einheiten

$$f_t(i) = \min_x\{0{,}5 \cdot (i + x - d) + c(x) + f_{t+1}(i + x - d)\}$$

$k = 3$

	0	1 (3+2+1+0.5)	2 (3+2+2+0.5*2)	3	4	$x_3{}^*$	$c_3{}^*(x)$
0	5+7=12	6,5+6=12,5	8+5=13	9,5+4=13,5	–	0	12
1	4+7=11	5,5+6=11,5	7+5=12	8,5+4=12,5	10+0=10	4	10
2	0+7=7	4,5+6=10,5	6+5=11	7,5+4=11,5	9+0=9	0	7
3	–	0,5+6=6,5	5+5=10	6,5+4=9,5	8+0=8	0	6,5
4	–	–	1+5=6	5,5+4=9,5	7+0=7	2	6

- $ft(i)$ 是在给定初始库存i的情况下，为了满足需求d（在这个月份是2单位），通过生产x单位产品而产生的总最小成本。
- x_3^* 显示了最优的生产量。
- $c_3^*(x)$ 显示了对应的总成本。

Rückschritt zum zweiten Monat

- Befriedigung der Nachfrage von 3 Einheiten

$$f_t(i) = \min_x \{0{,}5 \cdot (i + x - d) + c(x) + f_{t+1}(i + x - d)\}$$

$k = 2$

	0	1	2	3	4	$x_2{}^*$	$c_2{}^*(x)$
0	6+12=18	7,5+10=17,5	9+7=16	–	–	2	16
1	5+12=17	6,5+10=16,5	8+7=15	9,5+6,5=16	–	2	15
2	4+12=16	5,5+10=15,5	7+7=14	8,5+6,5=15	10+6=16	2	14
3	0+12=12	4,5+10=14,5	6+7=13	7,5+6,5=14	9+6=15	0	12
4	–	0,5+10=10,5	5+7=12	6,5+6,5=13	8+6=14	1	10,5

Rückschritt zum ersten Monat

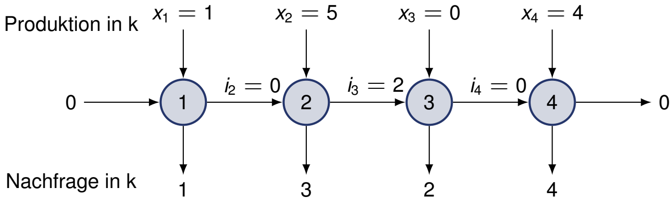
- Befriedigung der Nachfrage von 1 Einheiten

$$f_t(i) = \min_x \{0{,}5 \cdot (i + x - d) + c(x) + f_{t+1}(i + x - d)\}$$

$k = 1$

	0	1	2	3	4	${x_1}^*$	${c_1}^*(x)$
0	4+16=20	5,5+15=20,5	7+14=21	8,5+12=20,5	10+10,5=20,5	0	20

Lagerhaltungsproblem – Beispiel



Minimierung der Produktions-und Lagerkosten aus der Medizintechnikgeräteherstellung: 20.000 Euro

Lagerhaltungsproblem – Beispiel

Darstellung des Problems als Suche nach dem kürzesten Weg:

Kantengewichte entsprechen den Produktions- und Lagerkosten zwischen den Zuständen der einzelnen Stufen