Dynamisches Programmieren

Josef Fürst

(nach: Maniak, U. (2000) Wasserwirtschaft – Einführung in die Bewertung wasserwirtschaftlicher Vorhaben. Springer: Berlin – Heidelberg – New York. pp. 219-225.)

Optimierung der Abgaben aus Speichern

Das dynamische Programmieren (DP) ist kein direktes Optimierungsverfahren wie das lineare Programmieren, sondern eine Lösungsstrategie für mehrstufige Probleme wie sie z.B. beim Betrieb von Speichern auftreten. Beim DP wird ein mehrstufiges oder sequentielles Entscheidungsproblem, das viele miteinander verknüpfte Variable enthalten kann, in eine Folge von einstufigen Entscheidungen, von denen jede nur eine oder wenige Variable enthält, umgeformt *(Dekompositionsprinzip)*. Das Teilproblem jeder einzelnen Stufe kann mit einem beliebigen Optimierungsverfahren gelöst werden, wobei durch das dynamische Programmieren die Einzelaufgaben miteinander verknüpft werden (Bronstein, 1986). Das DP liefert effiziente Lösungen von komplexen vielschichtigen Problemen und wird benutzt, um z.B. die Berechnungszeit zu verkürzen. Erfolgt keine Trennung in Stufen, nimmt die Berechnungsdauer meist exponentiell mit der Anzahl der Variablen zu. Zur Lösung wird von folgendem operationellen Konzept ausgegangen (Abb. 1):

- Das zu optimierende Problem wird in Teilprobleme zerlegt. Die optimale Alternative jedes Teilproblems wird sequentiell herausgesucht, ohne dass vorab alle Kombinationsmöglichkeiten spezifiziert werden.
- Da die Optimierung auf Teilprobleme angewendet wird, werden nichtoptimale Kombinationen automatisch eliminiert.
- Die Teilprobleme sind untereinander in einer bestimmten Form verknüpft, so dass eine Optimierung von unmöglichen Kombinationen ausgeschlossen ist.

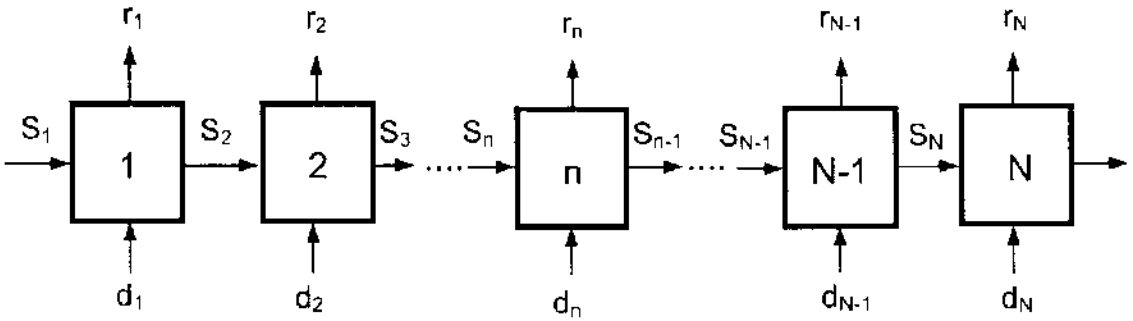

Abb. 1: Sequentielle Zerlegung des Optimierungsproblems in N Stufen beim dynamischen Programmieren

In jeder Stufe n des DP-Problems muss eine Entscheidung getroffen werden, insgesamt bei N Stufen. Die *Entscheidungsvariablen* d_n müssen für jede Stufe gesondert ermittelt werden. Eine Stufe kann mehr als eine Entscheidungsvariable enthalten. Die *Zustandsvariable* S_n beschreibt das System in der Stufe n. Sie kann diskret oder kontinuierlich sein. Die Zustandsvariablen S_n und S_{n+1} verbinden aufeinander folgende Stufen. Nachdem für eine Stufe die optimale Entscheidung d_n getroffen worden ist, ist diese daher auch zulässig für die gesamte verknüpfte Entscheidungskette (Abb. 1). Das skalare Maß für die *Wirksamkeit* r_n ist eine Funktion des In- und Output und der Entscheidungsvariablen, $r_n = r\ (S_n,\ S_{n+1},\ d_n)$. Der Zustandsübergang von einer Stufe zur nächsten wird durch die *Zustandstransformation* t_n ausgedrückt:

$$S_{n+1} = t_n\ (S_n,\ d_n) \qquad (1)$$

Für eine gegebene Stufe n ist das Ergebnis der Optimierung der noch verbleibenden Stufen unabhängig von dem, was bei vorangegangenen Stufen angewandt wurde.

Beim DP beginnt die Lösung mit der optimalen Entscheidung für die letzte Stufe und schreitet dann rückwärts zum Startpunkt. Die Rekursionsbeziehung beim Rückwärtsschreiten lautet (Abb. 1):

$$f_n^*(S_n) = \underset{d_n}{\text{opt}}\left[r_n(S_n, d_n) + f_{n+1}^*(S_{n+1})\right] = \underset{d_n}{\text{opt}}\left\{r_n(S_n, d_n) + f_{n+1}^*\left[t_n(S_n, d_n)\right]\right\} \quad \text{für n = 1 bis N-1} \tag{2}$$

Der algebraische Operator in Gl. 2 ist ein Pluszeichen. Es ist möglich, hierfür auch eine Subtraktion oder Multiplikation zu wählen. Mit f^* wird die optimale Entscheidung der zugehörigen Stufe bezeichnet.

Für eine vorwärts schreitende Rekursion wird entsprechend erhalten:

$$f_n^*(S_n) = \underset{d_n}{\text{opt}}\left[r_n(S_n, d_n) + f_{n-1}^*(S_{n-1})\right] \tag{3}$$

Die Rekursionsgleichung entspricht dem von Bellman entwickelten Prinzip der Optimalität. Es besagt, dass die Optimierung der letzten n Stufen eines dynamischen Programmierproblems nur vom Zustand S_n des Systems zu Beginn der n-ten Stufe abhängt und nicht von den Entscheidungen der vorhergehenden Stufen. Das Prinzip ist bei Speicherproblemen unter der Einschränkung gültig, dass das System Markov-Eigenschaften aufweist. Bei einem Markov-Modell 1. Ordnung hängt der Zustand nur von dem vorhergehenden Zustand ab, was bei monatlichen Abflüssen aus kleinen Gebieten der Fall sein kann.

Bei der monatlichen Wasserabgabe aus einem Speicher besteht oft das Problem, eine maximal verfügbare Wassermenge von b m^3 so aufzuteilen, dass der Erlös ein Maximum wird. Ist $r_i(x_i)$ der Erlös für die Wasserabgabe x_i im Monat i, so bildet die zu maximierende Zielgröße Z_{max} die Summe aus den maximierten Teilerlösen:

$$Z = \sum_{i=1}^{n} r_i(x_i) \Rightarrow \max \quad i = 1, \ldots, n$$

mit den Nebenbedingungen $\sum_{i=1}^{n} x_i \le b$

Die Zustandsvariable S_i beschreibt die Wassermenge, welche für die Monate i, i+l, ..., n noch verbleibt. Die Zustandsvariable S_i entspricht dem Betrag an Wasservolumen, der zu Beginn des Monats i verfügbar ist, abzüglich der Summe der Entscheidungen bis zum Monat i-1. In dieser Form ist das Problem zugänglich für eine Lösung durch das DP, wenn die DP-Funktion zur Transformation des Zustandes definiert wird (Gl. 1):

$$S_{i+1} = S_i - x_i = t_i(S_i, x_i) \quad \text{und} \quad S_i = b - \sum_{j=1}^{i-1} x_j \tag{4}$$

mit den Nebenbedingungen $S_1 = b$ und $S_n - x_n \ge 0$.

S_n kann durch folgende Gleichung ersetzt werden:

$$b - \sum_{j=1}^{n-1} x_j - x_n \ge 0 \quad \text{oder} \quad b - \sum_{j=1}^{n} x_j \ge 0 \quad \text{oder} \quad \sum_{j=1}^{n} x_j \le b \tag{5}$$

Zur Lösung des Problems wird bei der letzten (n-ten) Stufe begonnen:

$$\max_{x_n} r_n(x_n) \text{ so dass } S_n - x_n \ge 0 \tag{6}$$

Da der Wert von **S_n noch nicht bekannt ist, muss das einstufige Problem für alle in Betracht kommenden Werte von S_n**, d.h. $0 \le S_n \le b$ **gelöst werden**.

$$f_n^*(S_n) = \max_{x_n} r_n(x_n) \tag{7}$$

$f_n^*(S_n)$ = optimaler Erlös aus der Stufe n, wenn der Endzustand S_n ist.

Für jedes S_n muss die optimale Entscheidungsvariable x_n^* gefunden werden. Sie ist eine Funktion von S_n, d.h. $x_n(S_n)$

$$\max_{x_n \le S_n} r_n(x_n) = r_n(S_n) \text{ oder } f_n(S_n) = r_n(S_n) \tag{8}$$

Bei der nächsten Stufe nach rückwärts gehend sind nur 2 Stufen n und n-1 übrig für die Optimierung, d.h.

$$f_{n-1}^*(S_{n-1}) = \max_{x_{n-1}, x_n} \{r_{n-1}(x_{n-1}) + r_n(x_n)\} \tag{9}$$

Unter gewissen Bedingungen gilt:

$$f_{n-1}^*(S_{n-1}) = \max_{x_{n-1}} \left\{ r_{n-1}(x_{n-1}) + \max_{x_n} r_n(x_n) \right\} \tag{10}$$

$$f_{n-1}^*(S_{n-1}) = \max_{x_{n-1}} \{r_{n-1}(x_{n-1}) + f_n^*(S_n)\} \quad \text{mit} \quad S_n = S_{n-1} - x_{n-1} \text{ und } 0 < x_{n-1} \le S_{n-1}$$

oder allgemein für die i-te Stufe

$$f_i^*(S_i) = \max_{x_i} \{r_i(x_i) + f_{i+1}^*(S_{i+1})\} \tag{11}$$

$f_i^*(S_i)$ = optimaler Erlös der Stufe i, wenn zu Beginn der Stufe i der Anfangszustand S_i herrscht. Die optimale Entscheidung als Funktion von S_i ist $x_i(S_i)$.

Als ausreichende Bedingung muss $[r_{n-1}(x_{n-1}) + r_n(x_n)]$ eine wachsende Funktion von r_n sein für jeden möglichen Wert von r_{n-1}. Wird die Gleichung bis zur Stufe 1 gelöst, wird erhalten:

$$f_1(b) = \max Z = \sum_{i=1}^{n} r_i(x_i) \tag{12}$$

Damit kann die optimale Entscheidung x_i^* durch Vorwärtsrechnung gefunden werden

$$\begin{aligned}
x_1^* &= x_1(S_1) = x_1(b) \\
x_2^* &= x_2(b - x_1^*) = x_2(S_2^*) = x_2(S_1 - x_1^*) \\
x_3^* &= x_3(b - x_1^* - x_2^*) = x_3(S_3^*) = x_3(S_2^* - x_2^*) \\
&\vdots \\
x_n^* &= x_n\left(b - \sum_{i=1}^{n-1} x_i^*\right)
\end{aligned} \tag{13}$$

mit x_i^* = optimale Abgabe im Monat i.

Beispiel

Ein Speicher zur Wasserversorgung mit einem maximalen Speichervermögen von $3 \cdot 10^7$ m^3 (= 3 Einheiten) soll während eines Jahres das Wasser so abgeben, dass der Erlös zum Maximum wird. Der Zufluss QZ soll vierteljährlich betrachtet werden. Er beträgt im 1. Quartal 3 Einheiten, im 2. und 4. je eine Einheit und im 3. Quartal 2 Einheiten. Für jede abgegebene 1. Einheit beträgt der Erlös $2 \cdot 10^6$ G.E., für die 2. abgegebene Einheit $1.5 \cdot 10^6$ G.E. und für die 3. Einheit $1.0 \cdot 10^6$ G.E. (Erlös also 2, 3.5, bzw. $4 \cdot 10^6$ GE bei A = 1, 2, 3 Einheiten). Läuft bei gefülltem Speicher 1 Einheit über, tritt ein Schaden von $1.5 \cdot 10^6$ G.E. ein, beim Überlauf von 2 Einheiten $3.0 \cdot 10^6$ G.E. Wie muss die Abgabe A erfolgen, wenn der Füllstand am Anfang bzw. am Ende jeden beliebigen Wert annehmen kann?

Die Transformationsfunktion von einem Zustand zum folgenden wird durch die Wasserbilanz bestimmt:

$$\widetilde{S}_n = S_n + QZ_n - A_n$$

$\widetilde{S}_n = S(t_{n+1})$ Zustandsvariable am Ende der Stufe (Saison) n,

$S_n = S(t_n)$ Zustandsvariable am Anfang der Stufe (Saison) n.

QZ_n Zufluss während der Stufe i; $QZ_1 = 3$, $QZ_2 = 1$, $QZ_3 = 2$ und $QZ_4 = 1$ Einheit,

$A_n = d_n =$ Entscheidungsvariable, hier Abgabe;

Abgabe A Einheiten	Wirksamkeit r GE
1	2
2	2 + 1.5 = 3.5
3	2+1.5+1 =4.5
4	2+1.5+1-1.5=3.0
5	2+1.5+1-1.5-3=0
6	4.5-6.0=-1.5.

Infolge der vorgegebenen Quartalszuflüsse sind 4 Stufen zu unterscheiden. Die Rekursionsgleichung für den Rückwärtsschritt lautet:

$$f_n^*(S_n) = \max_{d_n}\left[r_n(S_n, d_n)\right] \text{ für } n = 4$$

$$f_n^*(S_n) = \max_{d_n}\left[r_n(S_n, d_n) + f_{n+1}^*(S_{n+1})\right] \text{ für } n = 1,2,3$$

Die Berechnung beginnt bei der letzten Stufe (Stufe 4) und erfolgt rückwärts bis zum Anfang (Stufe 1). Für jede Stufe wird das Maximum bestimmt.

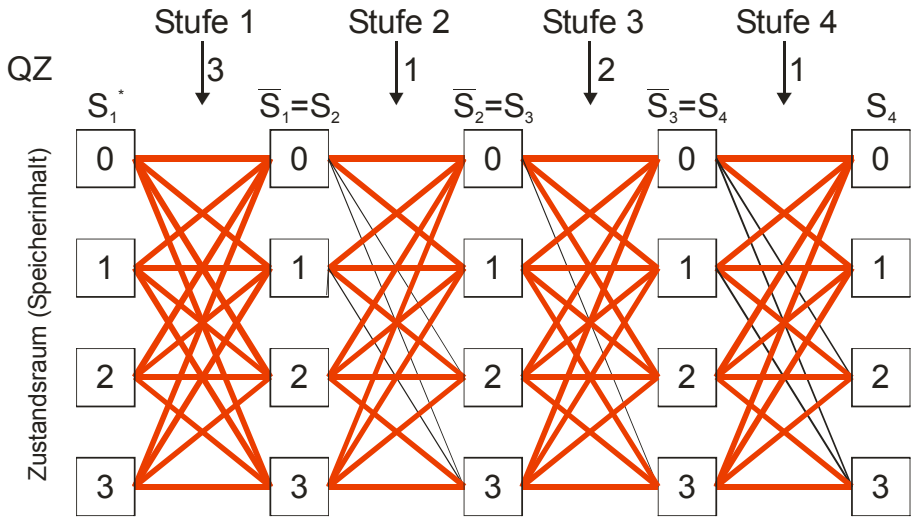

Abb. 2: Stufen und Zustandsraum beim DP für das Beispiel

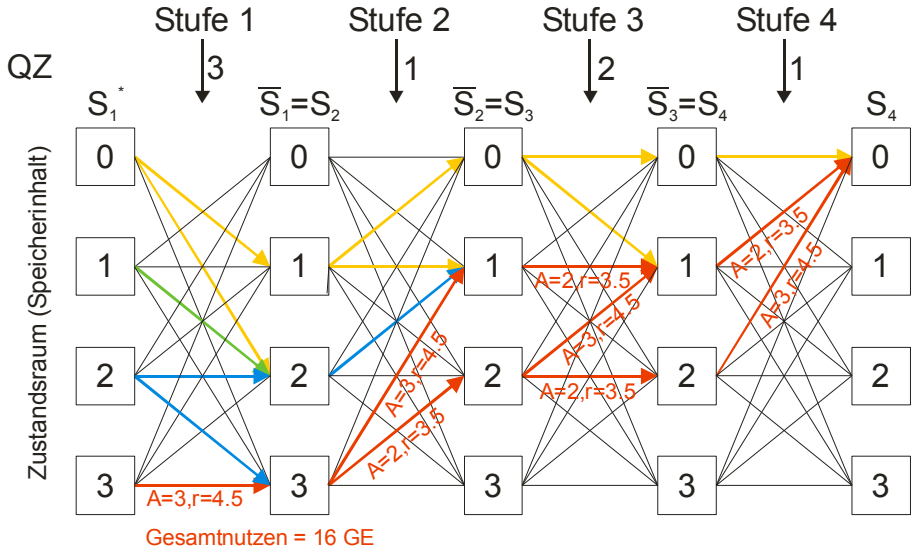

Abb. 3: Optimierte Abgaben bei der Vorwärtsrechnung

Tabelle: DP Berechnung für Stufen 4 bis 1 für einen Speicher mit einem maximalen Inhalt von 3 Volumseinheiten

Anfangs-speicherinhalt S_n	Gesamt-volumen S_n+QZ_n	Nutzen $f_n(S_n)=r_n(S_n,d_n)$ Endzustand Speichervolumen = 0	1	2	3	Max. Nutzen $f_n^*(S_n)$	Entscheid-ung (Abgabe) $d_n=A_n^*$	Endzu-stand Speicher S_n^*
Stufe 4	Nutzen $f_4(S_4) = r_4(S_4,d_4)$							
0	1							
1	2							
2	3							
3	4							
Stufe 3	Nutzen $f_3(S_3) = r_3(S_3,d_3) + f_4^*(S_4)$							
0	2							
1	3							
2	4							
3	5							
Stufe 2	Nutzen $f_2(S_2) = r_2(S_2,d_2) + f_3^*(S_3)$							
0	1							
1	2							
2	3							
3	4							
Stufe 1	Nutzen $f_1(S_1) = r_1(S_1,d_1) + f_2^*(S_2)$							
0	3							
1	4							
2	5							
3	6							

Eine Abgabe von 1 Einheit entspricht A = 2; also wird $f_4 (S_4) = r_4 (S_4 = 0, d_4 = A_4 = 1) = 2$ G.E.. Für $S_4 = 1$ und $S_4 = 0$ wird $A_4 = 1 + 1 - 0 = 2$ bzw. $f_4 (S_4) = r_4 (S_4 = 1, d_4 = A_4 = 0 = 3{,}5$ G.E. Die Berechnung erfolgt tabellarisch, wobei für alle Möglichkeiten des nutzbaren Wasserdargebots (hier: Speicherinhalt am Anfang einer Stufe plus Zufluss in die Stufe) und unter Beachtung der Bilanzgleichung die möglichen Abgaben berechnet werden (Tabelle). Die dritte bis 6. Spalte der Tabelle sind der Kern des dynamischen Programmierens. Aus den Spalten 3 bis 6 werden die Werte herausgesucht, welche den größten Nutzen f*(S) hervorrufen**.** Die zulässige Abgabe bzw. Speicherinhalte am Ende der Stufe werden in den beiden Spalten A^* und S^* notiert. Zum Gewinn jeder vorausgegangenen Stufe wird der maximale Wert von f^* addiert: $f_i (S_i) = r_i (S_i, d_i) + f^*_{i+1} (S_{i+1})$.

Nachdem alle Stufen rückwärts durchlaufen wurden, findet für einen beliebigen Anfangswert eine Vorwärtsberechnung statt. Wird als Startwert der Wert von $f_1^* = 16$ gewählt, erhält man $S_1 = 3$ mit einer optimalen Abgabe von $A_1 = 3$ und $S_1^* = S_2 = 3$. Für $S_2 = 3$ erhält man $f_2^* = 11.5$ und für $S_3 = 1$ wird $A_2 = 3$ bzw. für $S_3 = 2$ wird $A_2 = 2$.

Für die 3. Stufe kann von Speicherinhalten für $S_3 = 2$ und $S_3 = 1$ ausgegangen werden. Für $S_3 = 2$ ist $d_2 = 3$ oder 2 und für $S_4 = 1$ oder 2. Für $S_3 = 1$ ist $d_4 = A_4^* = 2$ und $S_4 = 1$. Für $S_4 = 1$ ist $A_2 = 2$ und $S_4 = 0$ und für $S_4 = 2$ wird $A_1 = 3$ und $S_4 = 0$ erhalten.