DINAMIČKO PROGRAMIRANJE

Branka MINICHREITER-KLEMENČIĆ*

1. UVOD

Savremeni ekonomista, inženjer ili praktičar iz neke druge oblasti, u rješavanju problematike iz svojeg područja gotovo se redovito susreće s problemom određivanja optimuma, odnosno optimalnih rješenja u datoj situaciji. Kvantitativne studije optimuma kao i metode njihovog određivanja obuhvaćene su teorijom optimizacije. Simbolička reprezentacija veza koje postoje među faktorima u problemu rezultira u matematičkom modelu, koji sadrži tri bitno različite komponente:[1])

1. Nezavisne varijable (varijable odluke), tj. faktore koji se mogu kontrolirati i mijenjati s ciljem postizanja optimuma;

2. Parametre, tj. faktore koje nije moguće kontrolirati;

3. Mjeru efektivnosti (vrijednost funkcije kriterija, funkcije cilja, funkcije koristi) pridruženu svakom mogućem rješenju problema. (Matematički izraz mjere efektivnosti je realna funkcija varijabli i parametara).

Postoji li više mogućih rješenja matematičkog modela, tj. više različitih skupova nezavisnih varijabli koje zadovoljavaju ograničenja definirana modelom, javlja se problem donošenja odluke, odluke o izboru najboljeg rješenja iz skupa svih mogućih rješenja. Optimalno rješenje jest rješenje za koje je pripadna vrijednost funkcije kriterija ekstremalna, maksimalna ili minimalna već prema naravi problema.

Danas je raspoloživ već čitav niz tehnika za određivanje optimalnih rješenja. Izbor tehnike optimizacije u rješavanju konkretnog matematičkog modela ovisi o njegovoj matematičkoj strukturi, njegovoj veličini, raspoloživim prikladnim tehnikama optimizacije, iskustvu analitičara itd. Jedna od sve više upotrebljavanih i usavršavanih tehnika je i metoda dinamičkog programiranja. Prema autoru početnih radova, prvih metoda kao i naziva — dinamičko programiranje — R. E. Bellmanu, teoretski izvori bili su mu rezultati J. Von Neumanna i A. Walda s područja teorije igara odnosno područja sekvencijalne analize.[2])

Dinamičko programiranje je tehnika razrađena za rješavanje klase problema koje općenito nazivamo višeetapnim procesima odlučivanja. Tu

*) Autor je magistar matematičkih nauka i asistent Fakulteta ekonomskih nauka u Zagrebu.

[1]) Za vrlo kratku i sažetu diskusiju matematičkih modela vidi ref. [24], str. 2—5.

[2]) Vidi ref. [1], str. 2.

klasu čine procesi u kojima se donosi niz odluka i one se donose postepeno u vremenu. Odluke nisu međusobno nezavisne već svaka općenito ovisi o nizu prethodnih odluka. Postoji naravno niz tipova problema koji traže postepeno, sekvencijalno donošenje odluka, ali se i u problemima koji ne sadrže vremensku komponentu eksplicite, može umjetno izvršiti dekompozicija na niz subproblema manjih dimenzija i tada postepeno vršiti optimizacija.

Pristup problemu pomoću dinamičkog programiranja sastoji se u biti u formuliranju višedimenzionalnog problema optimiziranja u formi niza jednodimenzionalnih kondicionalnih problema optimiziranja. U principu takva je reformulacija problema uvijek moguća i u tom smislu je dinamičko programiranje uvijek primjenljivo.

2. FORMULACIJA PROBLEMA DINAMIČKOG PROGRAMIRANJA

2.1. VIŠEETAPNI PROCES ODLUČIVANJA

Promatramo fizički sistem koji se u vremenu mijenja i na čiji tok možemo u većoj ili manjoj mjeri utjecati. U svakoj etapi razvoja procesa sistem se nalazi u nekom stanju karakteriziranom skupom parametara, tzv. varijabli stanja odnosno vektorom stanja. U svakoj etapi procesa vršimo izbor transformacije tog vektora u neki drugi vektor stanja tj. u sličan skup parametara s novim numeričkim vrijednostima i time djelujemo na smjer razvoja procesa. S procesom promjene sistema vezan je neki naš interes čiji »numerički ekvivalent« ovisi o odluci izabranoj u pojedinoj etapi. Problem je, dakle, kako organizirati neki proces, da bi s obzirom na postavljene zahtjeve i dozvoljene uslove efekt bio najbolji ili drugim riječima, problem je izvršiti optimalno planiranje nekog procesa. Postoje različiti kriteriji klasifikacije višeetapnih procesa odlučivanja. Najprirodnija i najopćenitija je podjela opće klase višeetapnih procesa odlučivanja na procese determinističkog i stohastičkog tipa. Zajedničko im je osnovno svojstvo da posljedice prethodnih odluka mogu biti upotrebljene za izbor budućih operacija, a bitna je razlika u tome što je u deterministčkom slučaju posljedica odluke izabrane iz skupa mogućih odluka u nekoj etapi potpuno određena tom odlukom, dok u stohastičkom slučaju posljedica odabrane odluke nastupa s izvjesnom vjerojatnošću.

2.2. MATEMATIČKA STRUKTURA VIŠEETAPNIH PROCESA ODLUČIVANJA

Matematičke strukturne karakteristike višeetapnih procesa odlučivanja dao je prvi S. Karlin [19]. Obuhvaćeni su procesi kod kojih je operator pomoću kojeg komponiramo ukupnu korist iz parcijalnih koristi od pojedinih etapa — zbrajanje.[3])

[3]) To naravno nije nužno u svakom slučaju, ali pretežno se u aplikacijama koristi takav tip funkcije ukupne koristi. Općenito, ukupna korist je neka funkcija individualnih koristi.

2.2.1. Deterministički slučaj

Neka je X prostor stanja, dakle prostor svih mogućih stanja u kojima se može nalaziti proces i neka je D prostor odluka, dakle prostor svih mogućih odluka koje možemo donijeti u nekoj etapi procesa.

Na produkt prostoru $X \times D$ definiramo nenegativnu funkciju koristi $r_i(x, d)$ koja za svako i izražava korist od procesa, ako smo u i-toj etapi na sistem koji je u stanju x primijenili odluku d. Svakoj odluci $d \in D$ odgovara transformacija $T_d \in \{T_d\}$ takva, da je $T_{d_{i-1}}(x_{i-1}) = x_i$ za svako $x_i \in X$.

Prostor politike ili prostor strategije definiramo kao $S = D \times D \times \ldots$, tako da svaki element $s \in S$, $s = (d_1, d_2 \ldots)$ predstavlja niz odluka ili jednu politiku. Neka je ukupna korist od procesa s početnim stanjem x i upotrebom politike s

$$R(x, s) = \sum r_i(x_i, d_i)$$

U većini slučajeva imamo $r_i(x_i, d_i) = r(x_i, d_i)$ za svako i, dakle

$$R(x, s) = \sum r(x_i, d_i).$$

Cilj nam je odrediti maksimalnu ukupnu korist za svako moguće početno stanje procesa. Svaka politika s^*, za koju $R(x, s)$ postiže maksimum, tj.

$$R(x, s^*) = \max_{s \in S} R(x, s)$$

je optimalna politika.

2.2.2. Stohastički slučaj

Elementi prostora stanja S su slučajne varijable s vrijednostima u nekom skupu U. Prostor odluka D sastoji se od skupa funkcija koji preslikavaju U na neki drugi prostor A. Funkcija koristi $r_i(u, a)$ je realna funkcija definirana za svako $u \in U$ i $a \in A$. Skup transformacija $\{T_d\}$ i prostor politike S definiran je kao i u deterministićkom slučaju.

Ako je ukupna korist od procesa, upotrebom politike s

$$R(x, s) = \sum E[r_i(x_i, d_i)]$$

pri čemu je

$E[r_i(x_i, d_i)]$ očekivana vrijednost slučajne varijable $r_i(x_i, d_i)$, optimalna politika s^* bit će ona za koju je

$$R(x, s^*) = \max_{s \in S} R(x, s)$$

2.3. PRINCIP OPTIMALNOSTI

Klasični pristup matematičkoj formulaciji problema pomoću dinamičkog programiranja bazira se na jednostavnom ali snažnom principu optimalnosti, koji glasi:[4])

— Optimalna politika ima svojstvo da kakvo god bilo početno stanje i početna odluka, preostale odluke moraju činiti optimalnu politiku s obzirom na stanje koje rezultira iz prve odluke.

Drugim riječima, u svakoj etapi treba uvijek tražiti optimalno nastavljanje procesa s obzirom na stanje u kojem se sistem nalazi u toj etapi.

U literaturi mogu se naći neke varijante i generalizacije principa optimalnosti, npr. »... bilo koji proces koji se odvija između dvije fiksne tačke mora se odvijati optimalno...«[5]), »...neka politika je optimalna onda i samo onda ako je ona optimalna u svakoj podetapi n-etapnog procesa...«[6]) ili »...proces ima svojstvo optimalnosti na nekoj domeni onda i samo onda ako ima to svojstvo na svakoj subdomeni te domene...«[7]).

U svakom slučaju to svojstvo optimalnosti politike je nužno, jer kad neki dio politike ne bi bio optimalan i cijela politika ne bi mogla biti optimalna.

Princip optimalnosti omogućuje nam formuliranje višeetapnog procesa odlučivanja u obliku funkcionalne rekurzivne jednadžbe, koja veže rješenje n-etapnog procesa s rješenjem (n+1)-etapnog procesa, pa kad nam je poznato neko inicijalno rješenje, tada upotrebom rekurzivne veze možemo naći rješenje problema s proizvoljnim brojem etapa.

2.4. MATEMATIČKA FORMULACIJA PROBLEMA DINAMIČKOG PROGRAMIRANJA

Pretpostavimo da proces počinje u stanju x_N i nakon N etapa završava u stanju x_0.

Neka je:

— d_i odluka u i-toj etapi primijenjena na proces u stanju x_i;

— $x_{i-1} = T_i(x_i, d_i)$ slijedeće stanje procesa;

— $r_i(x_i, d_i)$ korist pri prelazu procesa iz stanja x_i u stanje x_{i-1};

— R_N ukupna korist od N-etapnog procesa,

Proces prolazi kroz stanja[8])

$$x_N$$

$$x_{N-1} = T_{N-1}(x_N, d_N) \qquad (1)$$

$$\ldots$$

4) Usporediti ref. [2], str. 83.

5) Usporediti ref. [23], str. 610.

6) i 7) Usporediti ref. [12], str. 31.

8) Numeracija stanja $x_N, x_{N-1}, \ldots, x_0$ uzeta je iz formalnih razloga. Prirodna numeracija $x_0, \ldots, x_N$ dobije se naprosto renumeracijom stanja. Pogodnost upotrebljene numeracije jest da indeks i iznačuje broj etapa koje proces još treba proći.

$$x_0 = T_1 (x_1, d_1)$$

upravljan politikom $(d_N, \ldots, d_1)$, generirajući parcijalne koristi $r_N (x_N, d_N)$, $\ldots, r_1 (x_1, d_1)$.

Ukupna korist R_N, koja ovisi o nizu stanja kroz koja je proces prošao kao i o nizu odluka koje su bile primijenjene, općenito je neka funkcija individualnih koristi, tj.

$$R_N (x_N, x_{N-1}, \ldots, x_1, d_N, d_{N-1}, \ldots, d_1) =$$

$$= G \left[r_N (x_N, d_N), r_{N-1} (x_{N-1}, d_{N-1}), \ldots, r_1 (x_1 d_1) \right] \quad (2)$$

Koristeći relacije (1) u relaciji (2) možemo eliminirati stanja x_i $(i = 1 \ldots, N-1)$. Imamo

$$x_i = T_{i+1} (x_{i+1}, d_{i+1}) = T_{i+1} \left[T_{i+2} (x_{i+2}, d_{i+2}), d_{i+1} \right]$$

$$= T_{i+1} (x_{i+2}, d_{i+2}, d_{i+1}) = \ldots =$$

$$= T_{i+1} (x_N, d_N, d_{N-1}, \ldots, d_{i+1}).$$

Slijedi da je

$$R_N (x_N, d_N, d_{N-1} \ldots, d_1) =$$

$$= G \left[r_N (d_N, d_N), r_{N-1} (x_N, d_N, d_{N-1}), \ldots, r_1 (x_N, d_N, d_{N-1}, \ldots, d_1) \right]$$

to jest ukupna korist od N-etapnog procesa ovisi samo o početnom stanju x_N i politici $(d_N, d_{N-1}, \ldots, d_1)$.

Označimo li s

— $f_N (x_N)$ maksimalnu ukupnu korist od N-etapnog procesa započetog u stanju x_N,

tj. ukupnu korist od N etapa ako smo optimalnom politikom $(d_N^*, d_{N-1}^*, \ldots, d_1^*)$ proces proveli kroz stanja $x_N, x_{N-1}^*, \ldots, x_1^*$, tada je

$$f_N (x_N) = G \left[r_N (x_N, d_N^*), r_{N-1} (x_{N-1}^*, d_{N-1}^*), \ldots, r_1 (x_1^*, d_1^*) \right]$$

$$= \max G \left[r_N (x_N, d_N^*), r_{N-1} (x_{N-1}, d_{N-1}^*), \ldots, r_1 (x_1, d_1^*) \right]$$

$$f_N (x_N) = \max_{d_N, \ldots, d_1} G \left[r_N (x_N, d_N), r_{N-1} (x_{N-1}, d_{N-1}), \ldots, r_1 (x_1, d_1) \right] \quad (3)$$

uz uslov

$$x_{i-1} = T_i (x_i, d_i), \quad i = 1,2 \ldots, N,$$

ili

$$f_N (x_N) = \max_{d_N, \ldots, d_1} G \left[r_N (x_N, d_N), r_{N-1} (x_N, d_N, d_{N-1}), \ldots, r_1 (x_N, d_N, \ldots, d_1) \right] \quad (4)$$

Forma (3) sadrži N varijabli stanja, N varijabli odluka i N uslova, dok forma (4) sadrži N varijabli odluka i samo jednu varijablu stanja x_N.

Već u uvodu rekli smo da se bit pristupa pomoću dinamičkog programiranja sastoji u dekompoziciji višedimenzionalnog problema maksimiziranja na niz jednodimenzionalnih kondicionalnih problema maksimiziranja. Ako je to omogućeno, svaki subproblem sadrži jednu varijablu stanja i jednu varijablu odluke. Mogućnost dekompozicije ovisna je o tipu funkcije G.

L.G. Mitten [23] dao je 1964. godine dovoljan uslov, koji mora zadovoljavati funkcija G, da bi dekompozicija bila moguća. Prema njegovim rezultatima dovoljan uslov za dekompoziciju

$$\max_{d_N, d_{N-1}, \ldots, d_1} G \left[r_N (x_N, d_N), r_{N-1} (x_{N-1} d_{N-1}), \ldots, r_1 (r_1, d_1) \right] =$$

$$= \max_{d_N} G_1 \left\{ r_N (x_N, d_N), \max_{d_{N-1}, \ldots, d_1} G_2 [r_{N-1}, (x_{N-1}, d_{N-1}), \ldots, r_1 (x_1, d_1)] \right\}$$

jest 1. Separabilnost:

$$G \left[r_N (x_N, d_N), r_{N-1} (x_{N-1}, d_{N-1}), \ldots, r_1 (x_1 d_1) \right] =$$

$$= G_1 \left\{ r_N (x_N, d_N), G_2 \left[r_{N-1} (x_{N-1}, d_{N-1}), \ldots, r_1 (x_1, d_1) \right] \right\},$$

gdje su funkcije G_1 i G_2 realne, i

2. Monotonost:

G_1 je monotono nepadajuća funkcija od G_2 za svako r_N.

Ako je dakle uslov separabilnosti i monotonosti zadovoljen, tj.

$$R_N = r_N (x_N, d_N) \circ r_{N-1} (x_{N-1} d_{N-1}) \circ \ldots \circ r_1 (x_1 d_1) =$$

$$= r_N (x_N, d_N) \circ R_{N-1} \quad (5)$$

i

$$r_N (x_N, d_N) \circ \max_{d_{N-1}, \ldots, d_1} \left[r_{N-1} (x_{N-1} d_{N-1}) \circ \ldots \circ r_1 (x_1, d_1) \right] \geqslant$$

$$\geqslant r_N (x_N, d_N) \circ \left[r_{N-1} (x_{N-1} d_{N-1}) \circ \ldots \circ r_1 (x_1 d_1) \right] \quad (6)$$

ili

$$r_N (x_N, d_N) \circ f_{N-1} (x_{N-1}) \geqslant \quad (6')$$

$$\geqslant r_N (x_N, d_N) \circ \left[r_{N-1} (x_{N-1}, d_{N-1}) \circ \ldots \circ r_1 (x_1, d_1) \right],$$

pri čemu je „$\circ$" neki operator kompozicije, tada možemo izvesti opću rekurzivnu jednadžbu za funkciju $f_N (x_N)$ optimalne koristi N-etapnog procesa.

Po definiciji

$$f_N (x_N) = \max_{d_N, \ldots, d_1} G \left[r_N (x_N, d_N), r_{N-1} (x_{N-1}, d_{N-1}), \ldots, r_1 (x_1, d_1) \right]$$

$$= \max_{d_N, \ldots, d_1} \left[r_N (x_N, d_N) \circ r_{N-1} (x_{N-1}, d_{N-1}) \circ \ldots \circ r_1 (x_1, d_1) \right]$$

uz uslov $x_{i-1} = T_i (x_i, d_i), \quad i = 1,2, \ldots, N.$

Nadalje dobivamo

$$f_N(x_N) = \max_{d_N} \left\{ r_N(x_N, d_N) \circ \max_{d_{N-1}, \ldots, d_1} [r_{N-1}(x_{N-1}, d_{N-1}) \circ \ldots \circ r_1(x_1, d_1)] \right\}$$

$$= \max_{d_N} \left\{ r_N(x_N, d_N) \circ f_{N-1}(x_{N-1}) \right\}$$

uz uslov $x_{i-1} = T_i(x_i, d_i)$

$$= \max_{d_N} \left\{ r_N(x_N, d_N) \circ f_{N-1}[T_N(x_N, d_N)] \right\}$$

Ponovimo li postupak za $f_{N-1}(x_{N-1}), \ldots, f_2(x_2)$, dobivamo rekurzivne jednadžbe

$$f_i(x_i) = \max_{d_i} \begin{cases} r_i(x_i, d_i) \circ f_{i-1}[T_i(x_i, d_i)] \text{ za } i = 2, \ldots, N \\ r_i(x_i, d_i) \quad \text{za } i = 1 \end{cases} \tag{7}$$

Ako je operator komponiranja adicija, što će biti slučaj u svim našim primjerima, relacija (7) je oblika:

$$f_i(x_i) = \max_{d_i} \begin{cases} r_i(x_i, d_i) + f_{N-1}[T_i(x_i, d_i)] \text{ za } i = 2, \ldots, N \\ r_i(x_i, d_i) \quad \text{za } i = 1 \end{cases} \tag{8}$$

Do iste rekurzivne jednadžbe dinamičkog programiranja dolazimo upotrebom Bellmanovog principa optimalnosti. Takav pristup ilustrirat ćemo na primjerima.

3. TEHNIKE RJEŠAVANJA

3.1. OPĆA PROCEDURA RJEŠAVANJA

Nakon što smo odredili rekurzivnu relaciju za određeni problem, tj. odredili matematičku formu funkcija parcijalnih koristi i način njihovog komponiranja u funkciji ukupne koristi, cilj nam je odrediti funkciju ukupne koristi i funkciju optimalne politike.[9])

Znamo li neposredni efekt neke odluke d_i u prvoj etapi započetoj u stanju x_i i ukupni efekt od preostalih $(N-1)$ etapa procesa koji počinje u stanju $T_i(x_i, d_i)$, tada u stvari znamo ukupni efekt od odluke d_i na cijeli tok procesa. U i-toj etapi[10]), politika $(d_i, d_{i-1}, \ldots, d_1)$ koju treba izabrati u preostalom dijelu procesa, ne ovisi o prethodnoj politici $(d_N, \ldots, d_{i+1})$ kao takvoj, već samo o stanju x_i u kojem se proces nalazi radi upotrebljene politike $(d_N, \ldots, d_{i+1})$.

[9]) Vidjet ćemo kasnije, da metoda dinamičkog programiranja u rješenju daje *funkciju* optimalne politike, tj. optimalnu politiku kao funkciju početnog stanja procesa. Time ujedno dobivamo rješenje ne za jedan specijalni proces, nego za cijelu klasu procesa s istim karakteristikama a različitim početnim stanjima.

[10]) tj. u preostalih i etapa, prema našim oznakama.

Prema tome, ako se nalazimo u posljednoj etapi procesa, posljednja odluka d_1 ovisit će samo o stanju x_1 u kojem se proces nalazi radi neke upotrebljene politike $(d_N, \ldots, d_2)$ u prethodnih $(N-1)$ etapa procesa.

Za $i = 1$ relacija (8) svodi se na

$$f_1(x_1) = \max_{d_1} r_1(x_1, d_1).$$

Treba znači izabrati onu odluku d_1^*, koja maksimizira $r_1(x_1, d_1)$ tj. parcijalnu korist iz zadnje etape. Međutim, stanje x_1 nije poznato, jer ono je rezultat još nepoznate politike $(d_N, \ldots, d_2)$. Potrebno je dakle odrediti vrijednosti $f_1(x_1)$ za svako moguće stanje x_1, koje rezultira iz upotrebe svake moguće politike $(d_N, \ldots, d_2)$. Time imamo potpuno određenu optimalnu posljednju odluku d_1^* u procesu, iz svakog mogućeg stanja pretposljednje etape. Nastavljamo traženjem optimalne politike (d_2^*, d_1^*) za posljednje dvije etape. Kako znamo oblik funkcije neposredne parcijalne koristi $r_2(x_2, d_2)$, to (8) daje

$$f_2(x_2) = \max_{d_2} \left[r_2(x_2, d_2) + f_1(x_1) \right] \tag{9}$$

$$= \max_{d_2} \left\{ r_2(x_2, d_2) + f_1[T_2(x_2, d_2)] \right\}$$

Određujemo opet odluku d_2^* koja maksimizira desnu stranu od (9) za svako moguće stanje x_2, pri čemu je naredna optimalna odluka d_1^* već poznata za svako stanje $x_1 = T_2(x_2, d_2)$. Time dobivamo $f_2(x_2)$ i $(d_2^*, d_1^*) = d(x_2)$ dakle maksimalnu ukupnu korist iz dvoetapnog procesa koji počinje u stanju x_2 i pripadnu optimalnu politiku, koja također ovisi o početnom stanju x_2 dvoetapnog procesa.

Nastavljanjem iteracije dolazimo do

$$f_N(x_N) = \max_{d_N} \left\{ r_N(x_N, d_N) + f_{N-1}[T_N(x_N, d_N)] \right\}$$

dakle do funkcije maksimalne ukupne koristi i optimalne politike $(d_N^*, \ldots, d_1^*)$ za proces koji je započeo u stanju x_N i prošao N etapa.

Time je — teoretski — problem određivanja optimalnog rješenja za proces prikazan s (8) riješen.

Formulacija problema pomoću dinamičkog programiranja veoma je prikladna za rješavanje na elektronskom računskom stroju. Međutim, kod rješavanja problema većih dimenzija[11]), kao ozbiljno ograničenje javlja se kapacitet memorije. Ako radimo s analitičkim funkcijama možemo upotrebiti metode kalkulusa za određivanje maksimalne vrijednosti izraza u (8). Također iz analitičke strukture funkcije koju maksimiziramo, možemo ponekad zaključiti u kojem dijelu prostora odluke treba tražiti optimalnu odluku. Npr. ako je funkcija konkavna[12]) u cijelom području u kojem možemo varirati varijablu odluke, jasno je da će ekstremna vrijednost nastupiti na rubu područja.

[11]) Pod dimenzijom problema podrazumijevamo dimenziju vektora stanja.

[12]) U smislu, $y = y(x)$ je konkavna u intervalu $[x_1, x_2]$, ako je $y'' > 0$ u tom intervalu.

3.2. NEKE APROKSIMATIVNE METODE

Općenito funkcija f (x)[13]) pohranjena je u memoriji na mreži tačaka, tj. na konačnom diskretnom prikladno odabranom skupu. Takav način prikazivanja funkcije je idealan u smislu da je prikladan za kakvu god, proizvoljnu formu funkcije. No za izračunavanje vrijednosti funkcije s npr. deset dimenzija u samo 10 tačaka za svaku dimenziju već zahtjeva izračunavanje 10^{10} brojeva. Radi toga se kod funkcija za koje se može pretpostaviti barem neka pravilnost u njihovom (nepoznatom analitičkom) obliku primjenjuju prikladnije aproksimativne metode za rješavanje. Spomenimo npr. metodu aproksimacije pomoću polinoma [5]. Funkcija f (x) reprezentira se u formi

$$f(x) \approx \sum_{k=1}^{M} a_k \varphi_k(x),$$

gdje su $\varphi_k(x)$ elementarne funkcije, Legendreovi polinomi $P_k(x)$ ili polinomi Čebiševa $T_k(x)$ i pohranjuje u memoriji pomoću skupa od M koeficijenata $a_1, \ldots, a_M$. Na temelju rezultata provjeravanja efikasnosti takvih aproksimacija[14]) koja su vršena na nekoliko tipova funkcija kod kojih je bilo unaprijed poznato analitičko rješenje, nađeno je da se vrlo dobro slaganje na 2 ili više signifikantnih znamenaka dobiva s relativno niskim redom aproksimacije.

To je od izvanredno velike važnosti kod problema s većim brojem dimenzija. Takva reprezentacija ujedno olakšava problem eventualne interpolacije, koji je često potreban i koji je inače izvor velikih poteškoća u slučaju ako se funkcija upotrebljava u originalnom tabelarnom obliku.

Spomenimo i jednu od najnovijih metoda, »successive sweep method«, [25], S. R. Mc Reynoldsa. To je iterativna metoda za dobivanje rješenja u problemima optimalne kontrole. U biti ona je proširenje metode relaksacije drugog reda [22] na probleme optimalne kontrole s rubnim uslovima.

3.3. SNIŽAVANJE DIMENZIJE PROBLEMA

U problemima kod kojih se zahtjeva maksimizacija uz neka ograničenja može se izvršiti smanjivanje dimenzije problema, upotrebom koncepcije Lagrangeovih multiplikatora [8].[15]) Općenito za problem s M ograničenja, M — k njih može biti tretirano pomoću Lagrangeovih multiplikatora i dodano funkciji koristi. Time smanjujemo dimenziju problema i rješavamo problem s k varijabli stanja. Osim toga, pristup problemu pomoću Lagrangeovih multiplikatora omogućuje da problem, koji u originalnoj formi ima ograničenja, možemo riješiti metodama, koje inače nisu primjenjive na probleme s ograničenjima.

Klasična metoda Lagrangeovih multiplikatora primjenjiva je na diferencijabilne funkcije. Danas postoji generalizirana metoda Lagrangeovih multiplikatora [13], koja se može primijeniti na nediferencijabilne realne funkcije koristi uz proizvoljna ograničenja.

13) Govorimo općenito o nekoj funkciji optimalne koristi pa nije nužna oznaka broja etapa.

14) Izračunavanje je vršeno na IMB—7090.

15) Smatra se da je S. Dreyfus prvi došao na ideju primjene Lagrangeovih multiplikatora u cilju snižavanja dimenzije problema.

3.4. TEHNIKE TRAŽENJA

U proračunavanju rješenja kad je funkcija data tabelarno, potrebno je u svakoj etapi, za svaku moguću vrijednost varijabli stanja izračunati vrijednost funkcije koristi za svaku moguću odluku. Nakon toga za svako moguće stanje bira se optimalna vrijednost varijable odluke. Jasno je da je za problem s većim brojem varijabli stanja memorija stroja brzo iscrpljena, odnosno problem s većim brojem etapa zahtijeva vrlo mnogo vremena. Radi toga postavlja se pitanje možemo li maksimum funkcije locirati na efikasniji način. Ako se to uspije, traženje maksimuma nastavlja se samo unutar regije koja sadrži maksimum i time znatno odterećuje memorija stroja.

Za unimodalne funkcije jedne varijable nađeno je i dokazano [18] i [20], da je Fibonaccijevo traženje optimalna[16]) metoda za nalaženje maksimuma unimodalne funkcije unutar određenih granica tolerancije. Ona se ukratko sastoji u slijedećem:

Ako je data unimodalna funkcija f (x) nad intervalom $a_1 \leq x \leq b_1$ duljine I_1, definiramo

$$\Delta_2 = (F_{n-2}/F_n)\, I_1$$

i uzmemo da je

$$x_1 = a_1 + \Delta_2$$

$$x_2 = b - \Delta_2.$$

Ako je $f(x_1) \geq f(x_2)$, tada $a_2 = a_1$, $b_2 = x_2$ i $x_3 = b_2 - \Delta_3$, gdje je

$$\Delta_3 = (F_{n-3}/F_{n-1})\, I_2.$$

Proces nastavljamo dok maksimum od f (x) ne leži unutar intervala željene duljine $I_n = I_1 (F_0/F_n)$.

P. Krolak i L. Cooper [21] dali su tehniku za lociranje maksimuma unimodalne funkcije od više varijabli, koja se bazira na proširenju koncepcije Fibonaccijevog[17]) traženja na više dimenzija. Međutim, nije dosad pokazano da je to optimalna metoda traženja maksimuma u slučaju funkcije više varijabli. Ta tehnika može biti korištena i za funkcije koje nisu unimodalne ako se mogu razlučiti područja u kojima je ta funkcija unimodalna.

Dalji mogući način smanjivanja obima proračuna programa jest reduciranje mogućih vrijednosti za varijablu stanja. To je tzv. prilaz pomoću grube mreže (coarse grid approach). Upotrebljava se za redukciju broja mogućih vrijednosti svake varijable stanja. Ako je za neko određeno stanje x_n funkcija

$$r_n(x_n, d_n) + f_{n-1}[T_n(x_n, d_n)] \tag{10}$$

data na mreži vrijednosti varijable d_n i želimo odrediti

$$f_n(x_n) = \max_{d_n} \left\{ r_n(x_n, d_n) + f_{n-1}[T_n(x_n, d_n)] \right\}$$

16) Optimalna u smislu broja potrebnih računanja vrijednosti funkcije f (x).

17) Fibonacci je nadimak Leonarda iz Pise, koji je 1202. rješavajući »problem razmnažanja kunića« dobio rješenje u obliku niza $\{F_n\}$ generiranog rekurzivnom relacijom $F_n = F_{n-1} + F_{n-2}$, za $n \geq 2$, $F_0 = F_1 = 1$. Vidi ref. [27], str. 236.

tada računamo vrijednost od (10) najprije samo na vrlo gruboj mreži, samo na nekima od tačaka d_n na kojima je funkcija definirana. Rješenje te prve faze bit će gruba aproksimacija pravog rješenja, pa ako smo našli da se maksimum nalazi u tački $d_n = d_n^+$, tada dalje promatramo samo interval $[d_n^+ - \delta, d_n^+ + \delta]$ gdje je δ osnovni interval mreže na kojoj je funkcija definirana. U novoj, manjoj regiji mogućeg optimalnog rješenja, upotrebljavamo finiju mrežu i dobivamo na analogni način bolju aproksimaciju pravog rješenja. Postupak nastavljamo sve do željene pouzdanosti rješenja. Očito je da pri svakoj iteraciji možemo odbaciti neke, prije moguće, vrijednosti varijable odluke d_n, a time i neke moguće vrijednosti varijable stanja, jer prema transformaciji

$$x_{n-1} = T_n(x_n, d_n)$$

vidimo, da će broj mogućih vrijednosti varijable stanja x_{n-1} biti manji ukoliko je manji broj mogućih vrijednosti varijable odluke d_n.

Postoji uvijek opasnost da se promaši maksimum ako on nastupa nad uskim intervalom područja d_n; međutim, u praksi izgleda da takve krivulje nisu jako česte.

4. PRIMJENE

Karakteristična matematička struktura višeetapnih procesa odlučivanja javlja se gotovo kod svakog pokušaja realističkog tretiranja problema optimizacije. Radi toga dinamičko programiranje našlo je primjenu u na prvi pogled potpuno različitim oblastima, kao što su: ekonomija, logistika, matematička fizika itd. S područja ekonomije naročito mnogo je napravljeno na polju problema alokacije — npr. [1], [6], [9], zaliha — npr. [7], [10], [11], problema redoslijeda operacija — npr. [14], [16], [28], problema trgovačkog putnika — npr. [3], [15], itd.

Navođenje barem po jednog primjera iz širokog područja primjene metode dinamičkog programiranja zahtijevalo bi mnogo prostora. Zato su odabrana samo dva primjera: problem alokacije i problem zamjene opreme.

4.1. PROBLEM ALOKACIJE

Raspolažemo s početnom količinom sredstava S koju možemo upotrebiti u različitim aktivnostima. Rezultat upotrebe svih sredstava ili njihovog dijela u jednoj aktivnosti nazovimo korist od alokacije u tu aktivnost. Fundamentalni problem jest kako izvršiti raspodjelu ukupnih raspoloživih sredstava na pojedine aktivnosti da bi maksimizirali ukupnu korist od alokacije.

Neka je

— x_i količina sredstava dodijeljena i-toj aktivnosti

— $r_i(x_i)$ korist od dodjele sredstava x_i aktivnosti i.

Ako su aktivnosti međusobno nezavisne, ukupna korist od provedene alokacije jest

$$R(x_1, x_2, \ldots, x_N) = r_1(x_1) + r_2(x_2) + \ldots + r_N(x_N).$$

Treba odrediti

$$\max R(x_1, x_2, \ldots, x_N)$$

uz uslove

$$\sum_{i=1}^{N} x_i = S, \quad x_i \geq 0.$$

Rješavanje ovako definiranog modela alokacije zahtijeva upotrebu klasičnih metoda maksimiziranja funkcije više varijabli. Postoji niz opravdanih razloga[18]) za upotrebu efikasnijeg formuliranja istog problema. Razmotrimo pristup pomoću dinamičkog programiranja. Proces je statički, ali ga formalno dinamiziramo zamišljajući da se alokacije vrše jedna po jedna. Najprije alociramo sredstva x_N u N-tu aktivnost, zatim sredstva x_{N-1} u aktivnost (N—1). itd. Tražimo politiku $(x_N, \ldots, x_1)$ koja će maksimizirati ukupnu korist R.

Ako u prvoj etapi procesa alociramo sredstva x_N u N-tu aktivnost, korist od alokacije je $r_N(x_N)$ i za preostalih $(N-1)$ etapa ostaje $(S-x_N)$ sredstava. Ona se, slijedeći princip optimalnosti, moraju bez obzira na već izvršenu alokaciju x_N raspodijeliti na ostale aktivnosti tako da maksimiziraju ukupnu korist od alokacije u te aktivnosti. Dakle prema našim oznakama

$$f_N(S) = \max_{0 \leq x_N \leq S} \left[r_N(x_N) + f_{N-1}(S - x_N) \right]$$

za $N = 2, \ldots$

i

$$f_1(S) = \max r_1(S). \tag{11}$$

Npr. planira se raspodjela sredstava na dvije proizvodne grane za period od četiri godine.[19]) Raspolažemo s ukupnim sredstvima S_4. Alokacija x_i u prvu odnosno drugu granu rezultira u koristi $r_1(x_i) = x_i^2$ odnosno $r_2(x_i) = 2x_i^2$. Pri tome se sredstva djelomično troše i na kraju godine preostaje $0{,}75\,x_i$ odnosno $0{,}30\,x_i$ sredstava. Ukupna preostala sredstva na kraju godine čine početna sredstva za raspodjelu u slijedećoj godini. Stanje procesa karakterizirano je količinom sredstava raspoloživih za alokaciju i prema našim oznakama proces prolazi kroz stanja S_4, S_3, S_2, S_1, završava u stanju S_0 i vođen je politikom (x_4, x_3, x_2, x_1). Pri tome x_i označuje sredstva koja su dodijeljena prvoj grani ako je preostalo još i etapa. Sredstva koja se dodjeljuju drugoj grani određena su alokacijom u prvu granu, jer je

$$S_{i-1} = 0{,}75\,x_i + 0{,}3(S_i - x_i). \tag{12}$$

Koristeći relacije (11) i (12) dobivamo funkcije ukupne koristi

$$f_1(S_1) = \max_{0 \leq x_1 \leq S_1} \{ r_1(x_1) + r_2(S - x_1) \}$$

---

[18]) Detaljna diskusija poteškoća kod rješavanja ovakve formulacije problema data je npr. u ref. [4].

[19]) Usporediti ref. [26], str. 66.

$$f_2(S_2) = \max \{ r_1(x_2) + r_2(S_2 - x_2) + f_1 [0{,}75\, x_2 + 0{,}3\,(S_2 - x_2)] \}$$

$$0 \le x_2 \le S_2 \qquad (13)$$

$$f_3(S_3) = \max \{ r_1(x_3) + r_2(S_3 - x_3) + f_2 [0{,}75 x_3 + 0{,}3\,(S_3 - x_3)] \}$$

$$0 \le x_3 \le S_3$$

$$f_4(S_4) = \max \{ r_1(x_4) + r_2(S_4 - x_4) + f_3 [0{,}75\, x_4 + 0{,}3\,(S_4 - x_4)] \}$$

$$0 \le x_4 \le S_4$$

Koristeći činjenicu da su izrazi u (13) koje maksimiramo konkavne funkcije u području $0 \le x_i \le S_i$, dobivamo da je optimalna politika $(x_4, x_3, x_2, x_1) = (S_4, S_3, 0, 0)$. Uz početna sredstva S_4, maksimalna korist $f_4(S_4) = 2{,}26\, S_4^2$ na kraju četvrte godine preostaje $S_0 = 0{,}0504\, S_4$ sredstava, a optimalna politika alociranja (u prvu granu) jest $(x_4, x_3, x_2, x_1) = (S_4, S_3, 0, 0) = (S_4, 0{,}75\, S_4, 0, 0)$. Variranjem početnih sredstava S_4 dobivamo rješenje za cijelu klasu isto građenih problema koji imaju različita početna stanja.

4.2. PROBLEM ZAMJENE OPREME[20])

U proizvodnom procesu upotrebljavamo dvije mašine. Za godišnji prihod od procesa pretpostavljamo da je opadajuća funkcija starosti mašina i potrebno je od vremena do vremene zamijeniti svaku od mašina novom. Treba odlučiti kada zamijeniti mašine koje stare da bi napravili najbolji kompromis između troškova nabavke novih mašina i smanjenog prihoda koji rezultira iz upotrebe starih mašina. Na kraju svake godine postoji mogućnost zamjene prve, druge, obje ili niti jedne mašine. Problem je naći politiku zamjene koja maksimizira ukupni dohodak kroz N godina proizvodnje.

Stanje procesa definirajmo starošću obje mašine, dakle vektorom stanja

$$X(i) = \begin{bmatrix} x_1(i) \\ x_2(i) \end{bmatrix}.$$

Neka U_0 označuje odluku — ne zamjenjivati mašine;

U_1 — " — — zamijeniti prvu;

U_2 — " — — zamijeniti drugu;

U_3 — " — — zamijeniti obje.

Pretpostavimo nadalje da radi sigurnosti na poslu prva mašina mora biti zamijenjena novom poslije četiri, a druga poslije dvije godine upotrebe. Dakle, za vektor stanja X vrijedi ograničenje:

$$X(i) \le \begin{bmatrix} 3 \\ 1 \end{bmatrix}.$$

[20]) Usp. ref. [17], str. 25.

Transformacije vektora stanja kao rezultat odluka U_0, U_1, U_2, U_3 su slijedeće:

$$X(i+1) = T\,[x(i), U_0] = \begin{bmatrix} x_1(i)+1 \\ x_2(i)+1 \end{bmatrix}$$

$$X(i+1) = T\,[x(i), U_1] = \begin{bmatrix} 0 \\ x_2(i)+1 \end{bmatrix}$$

$$X(i+1) = T\,[x(i), U_2] = \begin{bmatrix} x_1(i)+1 \\ 0 \end{bmatrix}$$

$$X(i+1) = T\,[x(i), U_3] = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Neka se prihod u jednoj etapi, ovisno o stanju procesa kreće ovako:

Moguće stanje $X = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 2 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 3 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 3 \\ 1 \end{bmatrix}$
Prihod Y	10	10	5	3	8	7	4	1

Također su poznati troškovi zamjene mašine prvog tipa (6 jedinica) i mašine drugog tipa (4 jedinice).

Označimo li s

— $f_N(c_1, c_2)$ ukupni dohodak kroz N godina, ako smo upotrebili optimalnu politiku i pošli iz stanja $x_1 = c_1$, $x_2 = c_2$,

tada je

$$f_N(c_1, c_2) = \max_{U_0, U_1, U_2, U_3} \begin{cases} Y(c_1, c_2) + f_{N-1}(c_1 + 1, c_2 + 1) & \text{za } U_0 \\ Y(c_1, c_2) - 6 + f_{N-1}(0, c_2 + 1) & \text{za } U_1 \\ Y(c_1, c_2) - 4 + f_{N-1}(c_1 + 1, 0) & \text{za } U_2 \\ Y(c_1, c_2) - 10 + f_{N-1}(0, 0) & \text{za } U_3 \end{cases} \qquad (14)$$

uz restrikcije da kad je $c_1 = 3$, odluka mora biti U_1 ili U_3, odnosno kad je $c_2 = 1$ odluka mora biti U_2 ili U_3.

Relacija (14) vrijedi za $N = 2, \ldots$ a za $N = 1$ imamo inicijalni uslov

$$f_1(c_1, c_2) = Y(c_1, c_2) \qquad (15)$$

tj. ako proces traje samo jednu godinu, nećemo nabavljati novu mašinu pa je optimalna odluka U_0.

Uzmemo li npr. $N = 5$, $c_1 = 0$, $c_2 = 0$, dakle proces koji počinje s novim mašinama i traje 5 godina, relacije (14) i (15) daju optimalnu politiku (U_2, U_1, U_2, U_0, U_0), sistem prolazi kroz stanja $\begin{bmatrix}0\\0\end{bmatrix}$ $\begin{bmatrix}1\\0\end{bmatrix}$ $\begin{bmatrix}0\\1\end{bmatrix}$ $\begin{bmatrix}1\\0\end{bmatrix}$ $\begin{bmatrix}2\\1\end{bmatrix}$ i maksimalni ukupni dohodak je 28 jedinica.

(Rad primljen februara 1969.)

LITERATURA

1. R. E. Bellman, *An Introduction to the Theory of Dynamic Programming*, The RAND Corp., R-245, 1953.
2. R. E. Bellman, *Dynamic Programming*, Princeton Univ. Press, 1957.
3. R. E. Bellman, »Dynamic Programming treatment of the Travelling Salesman Problem«, Journal of the ACM, Vol. 9, 1962.
4. R. E. Bellman, S. Dreyfus, *Applied Dynamic Programming*, Princeton Univ. Press, 1962.
5. R. E. Bellman, R. Kalaba, B. Kotkin, »Polynomial Approximation — A New Technique in Dynamic Programming; Allocation Process,«, Math. of Comp., Vol. 17, 1963.
6. R. E. Bellman, W. Karush, »On a New Functional Transform in Analysis: The Maximum Transform, Bull. Am. Math. Soc., Vol. 67, 1961.
7. A. S. Cahn, H. M. Wagner, *Statistical Management of Inventory*, The RAND Corp., RM-3023-PR, 1962.
8. S. E. Dreyfus, »Computational Aspects of Dynamic Programming«, Oper. Res., Vol. 5, 1957.
9. S. E. Dreyfus, *Dynamic Programming Solution of Allocation Problems*, The RAND Corp., P-1083, 1957.
10. A. Dvoretzky, J. Kiefer, J. Wolfowitz, »The Inventory Problem I. Case of Known Distribution of Demand«, Econometrica, Vol. 20, 1952.
11. A. Dvoretzky, J. Kiefer, J. Wolfowitz, »The Inventory Problem II. Case of Unknown Distribution of Demand«, Econometrica, Vol. 20, 1952.
12. *Dynamic Programming Techniques*, Applied Research Laboratory, Sylvania Electric Product Inc., Mass., RADC-TDR-62-285, 1962.
13. H. Everet III., »Generalized Lagrange Multiplier Method for Solving Problems of Optimal Allocation of Resources«, Oper. Res., Vol. 11. 1963.
14. D. R. Fulkerson, I. L. Glicksberg, O. A. Gross, *A Production-line Assignment Problems*, The RAND Corp., RM-1102, 1953.
15. R. H. Gonzalez, *Solution of the Travelling Salesman Problem by Dynamic Programming on the Hypercube*, Techn. Report No. 18, 0. R. Center M. I. T., 1962.
16. M. Held, R. M. Karp »A Dynamic Programming Approach to Sequencing Problems«, Journal Soc. Ind. Appl. Math, Vol. 10, 1962.
17. O. R. L. Jacobs, *An Introduction to Dynamic Programming*, Chapman & Hall, Ltd., 1967.
18. S. M. Johnson, *Best Exploaration for Maximum is Fibonaccian*, The RAND Corp., P—856, 1956.
19. S. Karlin, »The Structure of Dynamic Programming Models«, Naval Research Logistics Quarterly, Vol. 2, 1955.

20. J. Kiefer, »Sequential Minimax Search for a Maximum,« Proc. Am. Math. Soc., Vol. 4, 1953,
21. P. Krolak, L. Cooper, »An Extention of Fibonacci Search to Several Variables«, Comm. of the ACM, Vol. 6, 1963.
22. C. W. Merriam III., *Optimization Theory and the Design of Feedback Control Systems*, Mc Graw-Hill, Inc., 1964.
23. G. L. Mitten, »Composition Principles for Synthesis of Optimal Multistage Processes«, Oper. Res., Vol. 12, 1964.
24. G. L. Nemhauser, *Introduction to Dynamic Programming*, J. Wiley & Sons, Inc., 1966.
25. S. R. Mc Reynolds, »The Successive Sweep Method and Dynamic Programing«, Journal of Math. Anal. Appl., Vol. 19, 1967.
26. E. S. Ventcelj, *Elementy dinamičeskogo programmirovanija, Izd. NAUKA*, 1964.
27. D. J. Wilde, C. S. Beightler, *Foundations of Optimization*, Prentice — Hall, Inc., 1967.
28. S. Zacks, *On a Pseudosolution to a Scheduling Problem*, Stanford Univ., Appl. Math. and Stat. Lab., Techn. Rep., No. 84, 1962.