12 Tabele de dispersie

Multe aplicaţii necesită o mulţime dinamică pentru care să se aplice numai operaţiile specifice pentru dicţionare INSEREAZĂ, CAUTĂ şi ŞTERGE. De exemplu, un compilator pentru un limbaj de programare întreţine o tabelă de simboluri, în care cheile elementelor sunt şiruri arbitrare de caractere ce corespund identificatorilor din limbaj. O tabelă de dispersie este o structură eficientă de date pentru implementarea dicţionarelor. Deşi căutarea unui element într-o tabelă de dispersie poate necesita la fel de mult timp ca şi căutarea unui element într-o listă înlănţuită – un timp $\Theta(n)$ în cazul cel mai defavorabil – în practică, dispersia funcţionează extrem de bine. Pe baza unor ipoteze rezonabile, timpul preconizat pentru căutarea unui element într-o tabelă de dispersie este $O(1)$.

O tabelă de dispersie este o generalizare a noţiunii mai simple de tablou. Adresarea directă într-un tablou foloseşte abilitatea noastră de a examina o poziţie arbitrară în tablou într-un timp $O(1)$. Secţiunea 12.1 discută în detaliu despre adresarea directă. Adresarea directă este aplicabilă în cazul în care ne putem permite să alocăm un tablou care are câte o poziţie pentru fiecare cheie posibilă.

Când numărul cheilor memorate efectiv este relativ mic faţă de numărul total de chei posibile, tabelele de dispersie devin o alternativă eficientă la adresarea directă într-un tablou, deoarece o tabelă de dispersie foloseşte în mod normal un tablou de mărime proporţională cu numărul de chei memorate efectiv. În loc să folosim direct cheia ca indice în tablou, indicele este *calculat* pe baza cheii. În secţiunea 12.2 sunt prezentate ideile principale, iar în secţiunea 12.3 se descrie cum pot fi calculaţi indicii din tablou pe baza cheilor, folosind funcţii de dispersie. Sunt prezentate şi analizate diferite variaţii ale temei de bază; ideea de bază este că dispersia reprezintă o tehnică extrem de eficientă şi practică; operaţiile de bază pentru dicţionare necesită, în medie, doar un timp $O(1)$.

12.1. Tabele cu adresare directă

Adresarea directă este o tehnică simplă care funcţionează bine atunci când universul U al cheilor este rezonabil de mic. Să presupunem că o aplicaţie necesită o mulţime dinamică în care fiecare element are o cheie aleasă dintr-un univers $U = \{0, 1, ..., m-1\}$, unde m nu este foarte mare. Vom presupune că nu există două elemente având aceeaşi cheie.

Pentru a reprezenta mulţimea dinamică folosim un tablou sau o ***tabelă cu adresare directă*** $T[0..m-1]$, în care fiecare poziţie sau ***locaţie*** corespunde unei chei din universul U. Figura 12.1 ilustrează această abordare; locaţia k referă elementul având cheia k din mulţime. Dacă mulţimea nu conţine un element cu cheia k, atunci $T[k] = \text{NIL}$.

Operaţiile pentru dicţionare sunt uşor de implementat.

ADRESARE-DIRECTĂ-CAUTĂ(T, k)
  returnează $T[k]$

ADRESARE-DIRECTĂ-INSEREAZĂ(T, x)
  $T[cheie[x]] \leftarrow x$

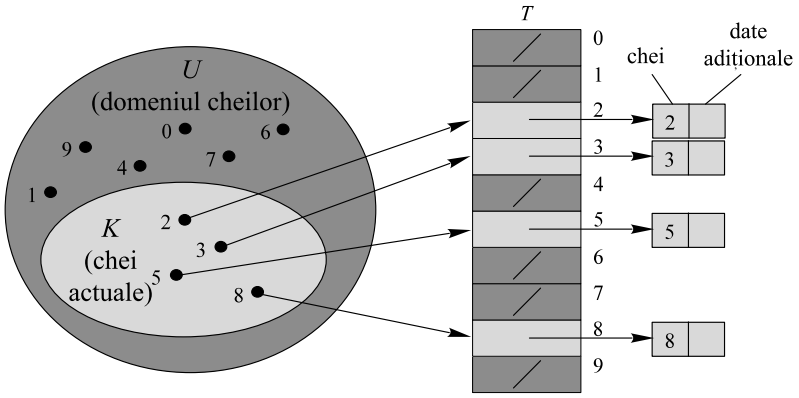

Figura 12.1 Implementarea unei mulţimi dinamice printr-un tablou cu adresare directă T. Fiecare cheie din universul $U = \{0, 1, ..., 9\}$ corespunde unui indice în tablou. Mulţimea $K = \{2, 3, 5, 8\}$ a cheilor efective determină locaţiile din tablou care conţin pointeri către elemente. Celelalte locaţii, haşurate mai întunecat, conţin NIL.

ADRESARE-DIRECTĂ-ŞTERGE(T, x)
  $T[cheie[x]] \leftarrow$ NIL

Fiecare dintre aceste operaţii este rapidă: este necesar doar un timp $O(1)$.

Pentru unele aplicaţii, elementele din mulţimea dinamică pot fi memorate chiar în tabela cu adresare directă. Aceasta înseamnă că în loc să memorăm cheia elementului şi datele adiţionale într-un obiect extern tabelei cu adresare directă, printr-un pointer dintr-o poziţie din tabelă către obiect, putem să memorăm obiectul în locaţia respectivă, economisind astfel spaţiu. Mai mult, deseori nu este necesar să memorăm câmpul cheie al obiectului, deoarece dacă avem indicele unui obiect în tabelă avem şi cheia sa. Oricum, dacă cheile nu sunt memorate, trebuie să existe o modalitate de a afirma că locaţia este goală.

Exerciţii

12.1-1 Se consideră o mulţime dinamică S care este reprezentată printr-o tabelă T cu adresare directă, de lungime m. Descrieţi o procedură care găseşte elementul maxim din S. Care este performanţa procedurii pentru cazul cel mai defavorabil?

12.1-2 Un ***vector de biţi*** este un tablou simplu de biţi (cu 0 şi 1). Un vector de biţi de lungime m ocupă mult mai puţin spaţiu decât un tablou de m pointeri. Descrieţi cum se poate folosi un vector de biţi pentru a reprezenta o mulţime dinamică având elemente distincte, fără date adiţionale. Operaţiile pentru dicţionare ar trebui să funcţioneze într-un timp $O(1)$.

12.1-3 Sugeraţi cum se poate implementa o tabelă cu adresare directă în care cheile elementelor memorate nu sunt neapărat distincte şi elementele pot avea date adiţionale. Toate cele trei operaţii pentru dicţionare (INSEREAZĂ, ŞTERGE şi CAUTĂ) ar trebui să se execute într-un timp $O(1)$. (Nu uitaţi că ŞTERGE are ca argument un pointer la obiectul ce va fi şters şi nu o cheie.)

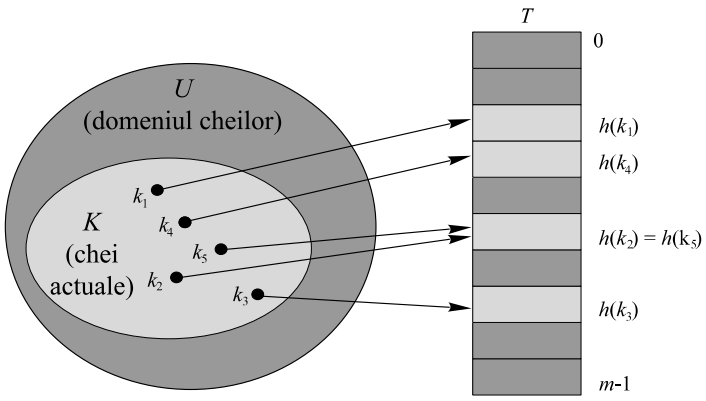

Figura 12.2 Folosirea unei funcţii de dispersie h pentru a transforma chei în poziţii din tabela de dispersie. Cheile k_2 şi k_5 se transformă în aceeaşi poziţie, deci sunt în coliziune.

12.1-4 $\star$ Dorim să implementăm un dicţionar folosind adresarea directă pe un tablou *uriaş*. La început, intrările în tablou pot conţine date nesemnificative, iar iniţializarea întregului tablou este nepractică datorită mărimii sale. Descrieţi o schemă pentru implementarea unui dicţionar cu adresare directă printr-un tablou uriaş. Fiecare obiect memorat ar trebui să utilizeze un spaţiu $O(1)$; operaţiile CAUTĂ, INSEREAZĂ şi ŞTERGE ar trebui să funcţioneze fiecare într-un timp $O(1)$; iniţializarea structurii de date ar trebui să necesite un timp $O(1)$. (*Indicaţie:* Folosiţi o stivă suplimentară, a cărei mărime să fie numărul de chei efectiv memorate în dicţionar, pentru a determina dacă o intrare dată în tabloul uriaş este validă sau nu.)

12.2. Tabele de dispersie

Dificultatea în adresarea directă este evidentă: dacă universul U este mare, memorarea tabelului T poate fi nepractică, sau chiar imposibilă, dată fiind memoria disponibilă a unui calculator uzual. Mai mult, mulţimea K a cheilor *efectiv memorate* poate fi atât de mică relativ la U, încât majoritatea spaţiului alocat pentru T ar fi irosit.

Când mulţimea K a cheilor memorate într-un dicţionar este mult mai mică decât universul U al cheilor posibile, o tabelă de dispersie necesită un spaţiu de memorie mult mai mic decât o tabelă cu adresare directă. Mai exact, cerinţele de memorare pot fi reduse la $\Theta(|K|)$, chiar şi atunci când căutarea unui element în tabela de dispersie necesită tot un timp $O(1)$. (Singurul punct slab este că această margine este stabilită pentru *timpul mediu*, în timp ce pentru adresarea directă ea era valabilă pentru *cazul cel mai defavorabil*.)

Prin adresare directă, un element având cheia k este memorat în locaţia k. Prin dispersie, acest element este memorat în locaţia $h(k)$; aceasta înseamnă că o ***funcţie de dispersie*** h este folosită pentru a calcula locaţia pe baza cheii k. În acest caz, h transformă universul U al cheilor în locaţii ale unei ***tabele de dispersie*** $T[0..m-1]$:

$$h : U \to \{0, 1, ..., m-1\}.$$

Vom spune că un element cu cheia k ***se dispersează*** în locaţia $h(k)$; de asemenea vom spune că

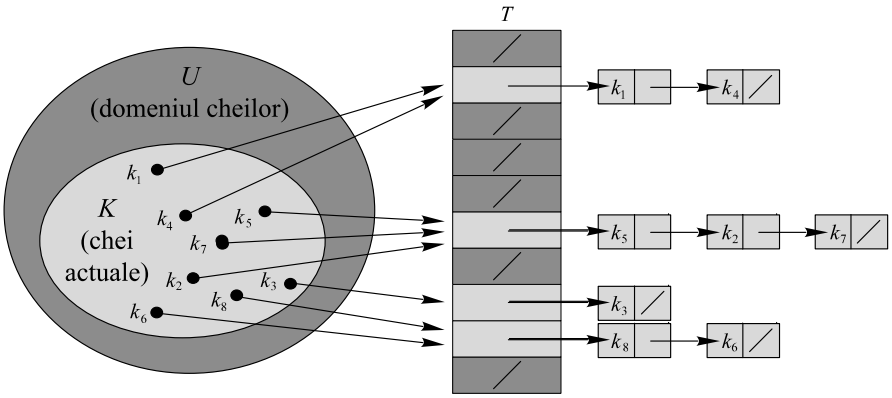

Figura 12.3 Rezolvarea coliziunii prin înlănţuire. Fiecare locaţie din tabela de dispersie $T[j]$ conţine o listă înlăţuită a tuturor cheilor a căror valoare de dispersie este j. De exemplu, $h(k_1) = h(k_4)$ şi $h(k_5) = h(k_2) = h(k_7)$.

$h(k)$ este ***valoarea de dispersie*** a cheii k. Figura 12.2 ilustrează ideea de bază. Scopul funcţiilor de dispersie este de a reduce domeniul indicilor tabloului care trebuie manipulaţi. În loc de $|U|$ valori, va trebui să manipulăm doar m valori. Cerinţele de memorare sunt reduse corespunzător.

Dezavantajul acestei idei frumoase este că două chei se pot dispersa în aceeaşi locaţie, adică se produce o ***coliziune***. Din fericire, există tehnici eficiente pentru rezolvarea conflictelor create de coliziuni.

Desigur, soluţia ideală ar fi să nu existe coliziuni. Putem încerca să atingem acest scop printr-o alegere potrivită a funcţiei de dispersie h. O idee este de a-l face pe h să pară "aleator", evitând astfel coliziunile sau cel puţin minimizând numărul lor. Termenul de "dispersie", care evocă imaginea unei fărâmiţări şi amestecări aleatoare, captează spiritul acestei abordări. (Bineînţeles, o funcţie de dispersie h trebuie să fie deterministă, în sensul că o intrare dată k trebuie să producă întotdeauna aceeaşi ieşire $h(k)$.) Deoarece $|U| > m$, există cu siguranţă două chei care să aibe aceeaşi valoare de dispersie; de aceea, evitarea totală a coliziunilor este imposibilă. Prin urmare, deşi o funcţie de dispersie "aleatoare", bine proiectată, poate minimiza numărul coliziunilor, avem în continuare nevoie de o metodă pentru rezolvarea coliziunilor ce apar.

Restul acestei secţiuni prezintă cea mai simplă tehnică de rezolvare a coliziunilor, numită înlănţuire. În secţiunea 12.4 se introduce o metodă alternativă de rezolvare a coliziunilor, numită adresare deschisă.

Rezolvarea coliziunii prin înlănţuire

Prin ***înlănţuire*** punem toate elementele ce se dispersează în aceeaşi locaţie, într-o listă înlănţuită, după cum se arată în figura 12.3. Locaţia j conţine un pointer către capul listei tuturor elementelor care se dispersează în locaţia j; dacă nu există astfel de elemente, locaţia j conţine NIL.

Operaţiile pentru dicţionare sunt uşor de implementat pe o tabelă de dispersie, în cazul în care coliziunile sunt rezolvate prin înlănţuire.

DISPERSIE-CU-INLĂNŢUIRE-INSEREAZĂ(T, x)
inserează x în capul listei $T[h(cheie[x])]$

DISPERSIE-CU-INLĂNŢUIRE-CAUTĂ(T, k)
caută un element cu cheia k în lista $T[h(k)]$

DISPERSIE-CU-INLĂNŢUIRE-ŞTERGE(T, x) indexDispersie-Cu-Inlantuire-Sterge@DISPERSIE-CU-INLĂNŢUIRE-ŞTERGE
şterge x din lista $T[h(cheie[x])]$

Timpul de execuţie pentru inserare în cazul cel mai defavorabil este $O(1)$. Pentru căutare, timpul de execuţie în cazul cel mai defavorabil este proporţional cu lungimea listei; vom analiza îndeaproape această situaţie în cele ce urmează. Ştergerea unui element x poate fi realizată într-un timp $O(1)$ dacă listele sunt dublu înlănţuite. (Dacă listele sunt simplu înlănţuite, atunci trebuie întâi să-l găsim pe x în lista $T[h(cheie[x])]$, astfel încât legătura *next* a predecesorului lui x să fie modificată corespunzător ca să-l "ocolească" pe x; în acest caz, ştergerea şi căutarea au în esenţă acelaşi timp de execuţie.)

Analiza dispersiei cu înlănţuire

Cât de bine funcţionează dispersia prin înlănţuire? În particular, cât durează căutarea unui element având o cheie dată?

Fiind dată o tabelă de dispersie T cu m locaţii ce memorează n elemente, vom defini ***factorul de încărcare*** α pentru T prin n/m, raport care reprezintă numărul mediu de elemente memorate într-o înlănţuire. Analiza noastră se bazează pe α; adică, ne imaginăm α ca fiind fix în timp ce n şi m tind la infinit. (Se observă că α poate fi mai mic, egal sau mai mare decât 1.)

Comportamentul, în cazul cel mai defavorabil, al dispersiei prin înlănţuire este slabă: toate cele n chei se dispersează în aceeaşi locaţie, creând o listă de lungime n. Timpul de căutare pentru cazul cel mai defavorabil este astfel $\Theta(n)$ plus timpul pentru calculul funcţiei de dispersie–cu nimic mai bun decât în cazul în care am fi folosit o listă înlănţuită a tuturor elementelor. Este clar că tabelele de dispersie nu sunt folosite pentru performanţa lor în cazul cel mai defavorabil.

Performanţa dispersiei în cazul mediu depinde de cât de bine distribuie (în medie) funcţia de dispersie h mulţimea cheilor ce trebuie memorate în cele m locaţii. În secţiunea 12.3 se discută aceste probleme, dar deocamdată vom presupune că orice element se poate dispersa în oricare din cele m locaţii cu aceeaşi probabilitate, independent de locul în care s-au dispersat celelalte elemente. Vom numi această ipoteză de lucru ***dispersie uniformă simplă***.

Presupunem că valoarea de dispersie $h(k)$ poate fi calculată într-un timp $O(1)$, astfel încât timpul necesar pentru a căuta un element având cheia k depinde liniar de lungimea listei $T[h(k)]$. Lăsând de o parte timpul $O(1)$ necesar pentru calculul funcţiei de dispersie şi pentru accesul la locaţia $h(k)$, să luăm în considerare numărul mediu de elemente examinate de algoritmul de căutare, adică, numărul de elemente din lista $T[h(k)]$ care sunt verificate pentru a vedea dacă cheia lor este egală cu k. Vom considera două cazuri. În primul caz, căutarea este fără succes: nici un element din tabelă nu are cheia k. În al doilea caz, căutarea găseşte cu succes un element având cheia k.

Teorema 12.1 Într-o tabelă de dispersie în care coliziunile sunt rezolvate prin înlănţuire, o căutare fără succes necesită, în medie, un timp $\Theta(1+\alpha)$, în ipoteza dispersiei uniforme simple.

Demonstraţie. În ipoteza dispersiei uniforme simple, orice cheie k se poate dispersa cu aceeaşi probabilitate în oricare din cele m locaţii. Astfel, timpul mediu pentru o căutare fără succes, pentru o cheie k, este timpul mediu pentru căutarea până la coada uneia din cele m liste. Deci, numărul mediu de elemente examinate într-o căutare fără succes este α şi timpul total necesar (incluzând timpul pentru calculul lui $h(k)$) este $\Theta(1+\alpha)$. ∎

Teorema 12.2 Într-o tabelă de dispersie în care coliziunile sunt rezolvate prin înlănţuire, o căutare cu succes necesită, în medie, un timp $\Theta(1+\alpha)$, în ipoteza dispersiei uniforme simple.

Demonstraţie. Presupunem că cheia care este căutată poate fi, cu aceeaşi probabilitate, oricare din cele n chei memorate în tabelă. De asemenea, presupunem că procedura DISPERSIE-CU-INLĂNŢUIRE-INSEREAZĂ inserează un nou element la sfârşitul unei liste şi nu la începutul ei. (Conform exerciţiului 12.2-3, timpul mediu pentru o căutare cu succes este acelaşi indiferent dacă noile elemente se inserează la începutul sau la sfârşitul listei.) Numărul mediu de elemente examinate într-o căutare cu succes este cu 1 mai mare decât numărul de elemente examinate în cazul în care elementul căutat a fost inserat (deoarece fiecare nou element se adaugă la sfârşitul listei). În concluzie, pentru a afla numărul mediu de elemente examinate, vom considera media celor n obiecte din tabelă, ca fiind 1 plus lungimea presupusă a listei în care se adaugă al i-lea element. Lungimea medie a listei respective este $(i-1)/m$ şi deci, numărul mediu de elemente examinate într-o căutare cu succes este:

$$\frac{1}{n}\sum_{i=1}^{n}\left(1+\frac{i-1}{m}\right) = 1+\frac{1}{nm}\sum_{i=1}^{n}(i-1) = 1+\left(\frac{1}{nm}\right)\left(\frac{(n-1)n}{2}\right) = 1+\frac{\alpha}{2}-\frac{1}{2m}\,.$$

Deci, timpul total necesar unei căutări cu succes (incluzând timpul pentru calculul funcţiei de dispersie) este $\Theta(2+\alpha/2-1/2m) = \Theta(1+\alpha)$. ∎

Ce semnificaţie are această analiză? Dacă numărul de locaţii din tabela de dispersie este cel puţin proporţional cu numărul de elemente din tabelă, avem $n = O(m)$ şi, prin urmare, $\alpha = n/m = O(m)/m = O(1)$. Deci, căutarea necesită, în medie, un timp constant. Deoarece inserarea necesită un timp $O(1)$, în cazul cel mai defavorabil (vezi exerciţiul 12.2-3) iar ştergerea necesită un timp $O(1)$ în cazul cel mai defavorabil, când listele sunt dublu înlănţuite, toate operaţiile pentru dicţionare pot fi efectuate în medie într-un timp $O(1)$.

Exerciţii

12.2-1 Presupunând că folosim o funcţie de dispersie aleatoare h pentru a dispersa n chei distincte într-un tablou T de dimensiune m, care este numărul mediu de coliziuni? Mai exact, care este cardinalul probabil al mulţimii $\{(x,y) : h(x) = h(y)\}$?

12.2-2 Ilustraţi inserarea cheilor 5, 28, 19, 15, 20, 33, 12, 17, 10 într-o tabelă de dispersie cu coliziunile rezolvate prin înlănţuire. Tabela are 9 locaţii, iar funcţia de dispersie este $h(k) = k \bmod 9$.

12.2-3 Argumentaţi că timpul mediu pentru o căutare cu succes prin înlănţuire este acelaşi indiferent dacă noile elemente se inserează la începutul, respectiv la sfârşitul unei liste. (*Indicaţie:* Arătaţi că timpul mediu pentru o căutare cu succes este acelaşi pentru *oricare* două ordine ale unei liste.)

12.2-4 Profesorul Marley emite ipoteza că se poate îmbunătăţi esenţial performanţa dacă modificăm schema de înlănţuire astfel încât fiecare listă să fie păstrată ordonată. În ce mod afectează modificarea profesorului timpul de execuţie pentru căutari cu succes, căutări fără succes, inserări şi ştergeri?

12.2-5 Sugeraţi cum poate fi alocat şi dealocat spaţiul de gestionare pentru elemente în cadrul tabelei de dispersie prin legarea tuturor locaţiilor nefolosite într-o listă liberă. Presupunem că o locaţie poate memora un "indicator" (engl. *flag*) împreună cu un element şi un pointer, sau cu doi pointeri. Toate operaţiile pentru dicţionare pentru lista liberă trebuie să funcţioneze într-un timp $O(1)$. Este necesar ca lista liberă să fie dublu înlănţuită sau este suficient să fie simplu înlănţuită?

12.2-6 Arătaţi că dacă $|U| > nm$, există o submulţime a lui U de mărime n ce conţine chei care se dispersează toate în aceeaşi locaţie, astfel încât timpul de căutare, pentru dispersie cu înlănţuire, în cazul cel mai defavorabil, este $\Theta(n)$.

12.3. Funcţii de dispersie

În această secţiune vom discuta unele aspecte legate de construirea de funcţii de dispersie bune, apoi vom prezenta trei scheme pentru crearea lor: dispersia prin diviziune, dispersia prin înmulţire şi dispersia universală.

Ce înseamnă o funcţie de dispersie bună?

O funcţie de dispersie bună satisface (aproximativ) ipoteza dispersiei uniforme simple: fiecare cheie se poate dispersa cu aceeaşi probabilitate în oricare dintre cele m locaţii. Mai formal, să presupunem că fiecare cheie este aleasă independent din U conform unei distribuţii de probabilitate P; deşi, $P(k)$ este probabilitatea de a fi aleasă cheia k. Atunci ipoteza dispersiei uniforme simple constă în

$$\sum_{k:h(k)=j} P(k) = \frac{1}{m} \text{ pentru } j = 0, 1, ..., m-1 \, . \tag{12.1}$$

Din păcate, în general, nu este posibilă verificarea acestei condiţii, deoarece de regulă distribuţia P nu este cunoscută.

Uneori (relativ rar) distribuţia P este cunoscută. De exemplu, presupunem că se cunoaşte faptul că cheile sunt numere reale aleatoare k, independent şi uniform distribuite în intervalul $0 \leq k < 1$. În acest caz se poate arăta că funcţia de dispersie

$$h(k) = \lfloor km \rfloor$$

satisface relaţia (12.1).

În practică, se pot folosi tehnici euristice pentru a crea funcţii de dispersie care par a se comporta bine. Informaţia calitativă despre P este uneori utilă în procesul de construcţie al lor. De exemplu, să considerăm o tabelă de simboluri a unui compilator, în care cheile sunt şiruri de caractere arbitrare, reprezentând identificatori dintr-un program. O situaţie des întâlnită este aceea în care simboluri foarte asemănătoare, ca `pt` şi `pts`, apar în acelaşi program. O funcţie de dispersie bună va minimiza posibilitatea ca asemenea variaţii să se disperseze în aceeaşi locaţie.

O abordare uzuală este de a obţine valoarea de dispersie într-un mod care se vrea independent de orice model sau şablon ce poate exista între date. De exemplu, "metoda diviziunii" (discutată în detaliu mai jos) calculează valoarea de dispersie ca fiind restul împărţirii cheii la un număr prim specificat. În afara cazului în care numărul este într-un fel dependent de şabloanele din distribuţia de probabilitate P, această metodă dă rezultate bune.

În final, să remarcăm faptul că unele aplicaţii ale funcţiilor de dispersie pot impune proprietăţi mai tari decât cele asigurate de dispersia uniformă simplă. De exemplu, am putea dori ca unele chei care sunt "apropiate" într-un anumit sens să producă valori de dispersie care să fie total diferite. (Această proprietate este dorită în special când folosim verificarea liniară, definită în secţiunea 12.4.)

Interpretarea cheilor ca numere naturale

Majoritatea funcţiilor de dispersie presupun universul cheilor din mulţimea $\mathbb{N} = \{0, 1, 2, ...\}$ a numerelor naturale. Astfel, dacă cheile nu sunt numere naturale, trebuie găsită o modalitate pentru a le interpreta ca numere naturale. De exemplu, o cheie care este un şir de caractere poate fi interpretată ca un întreg într-o bază de numeraţie aleasă convenabil. Prin urmare, identificatorul `pt` poate fi interpretat ca o pereche de numere zecimale (112,116), pentru că `p` = 112 şi `t` = 116 în mulţimea codurilor ASCII; atunci `pt` exprimat ca un întreg în baza 128 devine $(112 \cdot 128)+116 = 14452$. În mod obişnuit, în orice aplicaţie se poate crea direct o astfel de metodă simplă de a interpreta fiecare cheie ca un număr natural (posibil mare). În continuare, vom presupune că avem chei numere naturale.

12.3.1. Metoda diviziunii

Prin ***metoda diviziunii*** pentru crearea funcţiilor de dispersie, transformăm o cheie k într-una din cele m locaţii considerând restul împărţirii lui k la m. Prin urmare, funcţia de dispersie este

$$h(k) = k \mod m\,.$$

De exemplu, dacă tabela de dispersie are dimensiunea $m = 12$ şi cheia este $k = 100$, atunci $h(k) = 4$. Deoarece necesită doar o singură operaţie de împărţire, dispersia prin diviziune este rapidă.

Când folosim metoda diviziunii, de obicei evităm anumite valori ale lui m. De exemplu, m nu ar trebui să fie o putere a lui 2, pentru că dacă $m = 2^p$, atunci $h(k)$ reprezintă doar primii p biţi ai lui k. În afara cazului în care se cunoaşte apriori că distribuţia de probabilitate pe chei produce cu aceeaşi probabilitate oricare dintre şabloanele primilor p biţi, este mai bine să construim funcţia de dispersie ca dependentă de toţi biţii cheii. Puterile lui 10 ar trebui evitate dacă aplicaţia lucrează cu numere zecimale ca şi chei, pentru că în acest caz funcţia de dispersie

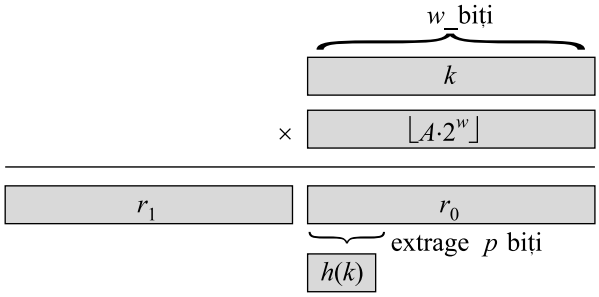

Figura 12.4 Metoda înmulţirii pentru dispersie. Reprezentarea pe w biţi a cheii k este înmulţită cu valoarea pe w biţi $\lfloor A \cdot 2^w \rfloor$, unde $0 < A < 1$ este o constantă aleasă convenabil. Primii p biţi ai celei de a doua jumătăţi a produsului, de lungime w biţi, formează valoarea de dispersie $h(k)$ dorită.

nu depinde de toate cifrele zecimale ale lui k. În final, se poate arăta că dacă $m = 2^p - 1$ şi k este un şir de caractere interpretat în baza 2^p, două şiruri de caractere care sunt identice, exceptând o transpoziţie a două caractere adiacente, se vor dispersa în aceeaşi valoare.

Valori bune pentru m sunt numerele prime nu prea apropiate de puterile exacte ale lui 2. De exemplu, să presupunem că dorim să alocăm o tabelă de dispersie, cu coliziunile rezolvate prin înlănţuire, pentru a reţine aproximativ $n = 2000$ şiruri de caractere, unde un caracter are 8 biţi. Nu ne deranjează să examinăm în medie 3 elemente într-o căutare fără succes, deci vom aloca o tabelă de dispersie de dimensiune $m = 701$. Numărul 701 este ales pentru că este un număr prim apropiat de $2000/3$, dar nu este apropiat de nici o putere a lui 2. Tratând fiecare cheie k ca un întreg, funcţia noastră de dispersie va fi

$$h(k) = k \mod 701 .$$

Ca o măsură de precauţie, putem să verificăm cât de uniform distribuie această funcţie de dispersie mulţimi de chei în locaţii, unde cheile sunt alese din date "reale".

12.3.2. Metoda înmulţirii

Metoda înmulţirii pentru crearea de funcţii de dispersie operează în doi paşi. În primul pas, înmulţim cheia k cu o constantă A din intervalul $0 < A < 1$ şi extragem partea fracţionară a lui kA. Apoi, înmulţim această valoare cu m şi considerăm partea întreagă inferioară a rezultatului. Pe scurt, funcţia de dispersie este

$$h(k) = \lfloor m(kA \mod 1) \rfloor ,$$

unde "$kA \mod 1$" înseamnă partea fracţionară a lui kA, adică, $kA - \lfloor kA \rfloor$.

Un avantaj al metodei înmulţirii este că valoarea lui m nu este critică. De obicei o alegem ca fiind o putere a lui 2 – $m = 2^p$ pentru un întreg p – pentru că atunci putem implementa funcţia pe majoritatea calculatoarelor după cum urmează. Să presupunem că lungimea cuvântului maşinii este de w biţi şi că k încape pe un singur cuvânt. Referindu-ne la figura 12.4, înmulţim întâi k cu întregul pe w biţi $\lfloor A \cdot 2^w \rfloor$. Rezultatul este o valoare pe $2w$ biţi $r_1 2^w + r_0$, unde r_1 este primul cuvânt al produsului şi r_0 este al doilea cuvânt al produsului. Valoarea de dispersie de p biţi dorită constă din cei mai semnificativi p biţi ai lui r_0.

Deşi această metodă funcţionează cu orice valoare a constantei A, ea lucrează mai bine cu anumite valori decât cu altele. Alegerea optimă depinde de caracteristicile datelor care sunt dispersate. Knuth [123] discută alegerea lui A în detaliu şi sugerează că

$$A = (\sqrt{5} - 1)/2 \approx 0.6180339887... \tag{12.2}$$

se va comporta relativ bine.

De exemplu, dacă avem $k = 123456, m = 10000$ şi A din relaţia (12.2), atunci

$$\begin{aligned} h(k) &= \lfloor 10000 \cdot (123456 \cdot 0.61803... \bmod 1) \rfloor = \lfloor 10000 \cdot (76300.0041151... \bmod 1) \rfloor = \\ &= \lfloor 10000 \cdot 0.0041151... \rfloor = \lfloor 41.151... \rfloor = 41 \, . \end{aligned}$$

12.3.3. Dispersia universală

Dacă cheile ce se dispersează sunt alese de un adversar maliţios, atunci el poate alege n chei care toate se vor dispersa în aceeaşi locaţie, rezultând un timp mediu de acces $\Theta(n)$. În cazul cel mai defavorabil, orice funcţie de dispersie fixată este vulnerabilă la acest tip de comportament; singura modalitate eficientă de a îmbunătăţi această situaţie este de a alege funcţia de dispersie în mod *aleator* astfel încât alegerea să fie *independentă* de cheile care se vor memora. Această abordare, numită ***dispersie universală***, produce în medie o performanţă bună, indiferent de ce chei sunt alese de către adversar.

Ideea care stă la baza dispersiei universale este de a selecta funcţia de dispersie în mod aleator în momentul execuţiei dintr-o clasă de funcţii construită cu atenţie. Ca şi în cazul sortării rapide, randomizarea asigură că nici o intrare nu va produce comportamentul în cazul cel mai defavorabil. Datorită randomizării, algoritmul se poate comporta diferit la fiecare execuţie, chiar pentru o aceeaşi intrare. Această abordare asigură o performanţă bună pentru cazul mediu, indiferent de cheile care sunt date ca intrare. Revenind la exemplul cu tabela de simboluri a unui compilator, descoperim că alegerea identificatorilor de către programator nu poate cauza, din punct de vedere al consistenţei, performanţe slabe la dispersie. Performanţa slabă apare doar atunci când compilatorul alege o funcţie de dispersie aleatoare care provoacă o dispersie slabă a mulţimii identificatorilor, dar probabilitatea ca această situaţie să apară este mică şi este aceeaşi pentru orice mulţime de identificatori de aceeaşi mărime.

Fie $\mathcal{H}$ o colecţie finită de funcţii de dispersie care transformă un univers dat U al cheilor, în domeniul $\{0, 1, ..., m-1\}$. O astfel de colecţie se numeşte ***universală*** dacă pentru fiecare pereche de chei distincte $x, y \in U$, numărul de funcţii de dispersie $h \in \mathcal{H}$ pentru care $h(x) = h(y)$ este exact $|\mathcal{H}|/m$. Cu alte cuvinte, cu o funcţie de dispersie aleasă aleator din $\mathcal{H}$, şansa unei coliziuni între x şi y când $x \neq y$ este exact $1/m$, care este exact şansa unei coliziuni dacă $h(x)$ şi $h(y)$ sunt alese aleator din mulţimea $\{0, 1, ..., m-1\}$.

Următoarea teoremă arată că o clasă universală de funcţii de dispersie dă un comportament bun în cazul mediu.

Teorema 12.3 Dacă h este aleasă dintr-o colecţie universală de funcţii de dispersie şi este folosită pentru a dispersa n chei într-o tabelă de dimensiune m, unde $n \leq m$, numărul mediu de coliziuni în care este implicată o cheie particulară x, este mai mic decât 1.

Demonstraţie. Pentru fiecare pereche y, z de chei distincte, fie c_{yz} variabila aleatoare care are valoarea 1 dacă $h(y) = h(z)$ (adică, dacă y şi z sunt în coliziune folosind h) şi 0 în caz contrar.

Deoarece, prin definiţie, o singură pereche de chei sunt în coliziune cu probabilitatea $1/m$, avem

$$E[c_{yz}] = 1/m \, .$$

Fie C_x numărul total de coliziuni în care este implicată cheia x dintr-o tabelă de dispersie T de dimensiune m conţinând n chei. Din relaţia (6.24) obţinem

$$E[C_x] = \sum_{\substack{y \in T \\ y \neq x}} E[c_{xy}] = \frac{n-1}{m} \, .$$

Deoarece $n \leq m$, obţinem $E[C_x] < 1$. ∎

Dar cât de uşor este să construim o clasă universală de funcţii de dispersie? Este relativ uşor, după cum se poate demonstra folosind unele cunoştinţe din teoria numerelor. Să alegem dimensiunea m a tabelei noastre ca fiind număr prim (ca şi în metoda diviziunii). Descompunem o cheie x în $r+1$ octeţi (de exemplu, caractere sau subşiruri binare de caractere de lungime fixă), astfel încât $x = \langle x_0, x_1, ...x_r \rangle$; singura cerinţă este ca valoarea maximă a unui octet să fie mai mică decât m. Notăm prin $a = \langle a_0, a_1, ...a_r \rangle$ un şir de $r+1$ elemente alese aleator din mulţimea $\{0, 1, ..., m-1\}$. Definim o funcţie de dispersie corespunzătoare $h_a \in \mathcal{H}$ astfel:

$$h_a(x) = \sum_{i=0}^{r} a_i x_i \mod m \, . \tag{12.3}$$

Cu această definiţie,

$$\mathcal{H} = \bigcup_{a} \{h_a\}. \tag{12.4}$$

are m^{r+1} elemente.

Teorema 12.4 Clasa $\mathcal{H}$ definită de relaţiile (12.3) şi (12.4) este o clasă universală de funcţii de dispersie.

Demonstraţie. Considerăm o pereche oarecare de chei distincte x şi y. Presupunem că $x_0 \neq y_0$. (O argumentaţie similară se poate face pentru diferenţa dintre oricare alte poziţii ale octeţilor.) Pentru orice valori fixe ale lui $a_1, a_2, ..., a_r$ există exact o valoare a lui a_0 care satisface ecuaţia $h(x) = h(y)$; acest a_0 este soluţia ecuaţiei

$$a_0(x_0 - y_0) \equiv -\sum_{i=1}^{r} a_i(x_i - y_i) \pmod{m} \, .$$

Pentru a observa această proprietate să remarcăm faptul că deoarece m este prim, cantitatea nenulă $x_0 - y_0$ are un invers faţă de înmulţirea modulo m şi astfel există o soluţie unică pentru a_0 modulo m. (Vezi secţiunea 33.4.) Prin urmare, fiecare pereche de chei x şi y este în coliziune pentru exact m^r valori ale lui a, deoarece ea este în coliziune exact o dată pentru fiecare valoare posibilă a lui $\langle a_1, a_2, ..., a_r \rangle$ (de exemplu, pentru valoarea unică a lui a_0 observată mai sus). Deoarece există m^{r+1} valori posibile pentru secvenţa a, cheile x şi y sunt în coliziune exact cu probabilitatea $m^r/m^{r+1} = 1/m$. În concluzie, $\mathcal{H}$ este universală. ∎

Exerciţii

12.3-1 Presupunem că dorim să efectuăm o căutare într-o listă înlănţuită de lungime n, în care fiecare element conţine o cheie k împreună cu o valoare de dispersie $h(k)$. Fiecare cheie este un şir lung de caractere. Cum putem profita de valorile de dispersie când căutăm în listă un element cu o cheie dată?

12.3-2 Presupunem că un şir de r caractere este dispersat pe m poziţii, fiind considerat ca număr în baza 128, căruia i se aplică metoda diviziunii. Numărul m este uşor de reprezentat în memorie ca un cuvânt pe 32 de biţi, dar şirul de r caractere, tratat ca un număr în baza 128, necesită multe cuvinte. Cum putem aplica metoda diviziunii pentru a calcula valoarea de dispersie a şirului de caractere fără a folosi mai mult decât un număr suplimentar constant de cuvinte de memorie în afară de şirul propriu-zis?

12.3-3 Se consideră o versiune a metodei diviziunii în care $h(k) = k \mod m$, unde $m = 2^p - 1$ şi k este un şir de caractere interpretat în baza 2^p. Arătaţi că dacă şirul de caractere x poate fi obţinut din şirul de caractere y prin permutări de caractere, atunci x şi y se dispersează în aceeaşi valoare. Daţi un exemplu de aplicaţie în care această proprietate nu este dorită pentru o funcţie de dispersie.

12.3-4 Se consideră o tabelă de dispersie de dimensiune $m = 1000$ şi funcţia de dispersie $h(k) = \lfloor m(kA \mod 1) \rfloor$ pentru $A = (\sqrt{5} - 1)/2$. Calculaţi locaţiile în care se pun cheile 61, 62, 63, 64, şi 65.

12.3-5 Arătaţi că dacă punem restricţia ca fiecare componentă a_i din a din relaţia (12.3) să fie nenulă, atunci mulţimea $\mathcal{H} = \{h_a\}$ definită ca în relaţia (12.4) nu este universală. (*Indicaţie:* Se consideră cheile $x = 0$ şi $y = 1$.)

12.4. Adresarea deschisă

Prin ***adresare deschisă***, toate elementele sunt memorate în interiorul tabelei de dispersie. Prin urmare, fiecare intrare în tabelă conţine fie un element al mulţimii dinamice, fie NIL. Când căutăm un element, vom examina sistematic locaţiile tabelei fie până când este găsit elementul dorit, fie până când este clar că elementul nu este în tabelă. Nu există liste sau elemente memorate în afara tabelei, aşa cum se întâmplă în cazul înlănţuirii. Prin urmare, prin adresare deschisă, tabela de dispersie se poate "umple" astfel încât nici o inserare nu mai este posibilă; factorul de încărcare α nu poate depăşi niciodată valoarea 1.

Desigur, am putea memora listele înlănţuite pentru a crea legăturile în cadrul tabelei de dispersie, în locaţiile altfel neutilizate ale acesteia (vezi exerciţiul 12.2-5), dar avantajul adresării deschise este că evită total folosirea pointerilor. Secvenţa de locaţii care se examinează nu se determină folosind pointerii, ci se *calculează*. Spaţiul de memorie suplimentar, eliberat prin faptul că nu memorăm pointerii, oferă tabelei de dispersie un număr mai mare de locaţii pentru acelaşi spaţiu de memorie, putând rezulta coliziuni mai puţine şi acces mai rapid.

Pentru a realiza inserarea folosind adresarea deschisă, examinăm succesiv sau ***verificăm*** tabela de dispersie până când găsim o locaţie liberă în care să punem cheia. În loc să fie fixat în ordinea $0, 1, ..., m - 1$ (care necesită un timp de căutare $\Theta(n)$), şirul de poziţii examinate

depinde de cheia ce se inserează. Pentru a determina ce locaţii sunt verificate, extindem funcţia de dispersie astfel încât ea să includă şi numărul de verificare (începând de la 0) ca un al doilea argument. Astfel, funcţia de dispersie devine

$$h : U \times \{0, 1, ..., m-1\} \to \{0, 1, ..., m-1\} .$$

În cadrul adresării deschise, cerem ca pentru fiecare cheie k, ***secvenţa de verificare***

$$\langle h(k, 0), h(k, 1), ..., h(k, m-1)\rangle$$

să fie o permutare a lui $\langle 0, 1, ..., m-1\rangle$, astfel încât fiecare poziţie din tabela de dispersie să fie considerată ca o eventuală locaţie pentru o nouă cheie pe măsură ce se umple tabela. În algoritmul următor, presupunem că elementele din tabela de dispersie T sunt chei, fără informaţii adiţionale; cheia k este identică cu elementul care conţine cheia k. Fiecare locaţie conţine fie o cheie, fie NIL (dacă locaţia este liberă).

DISPERSIE-INSEREAZĂ(T, k)
1: $i \leftarrow 0$
2: **repetă**
3: $\quad j \leftarrow h(k, i)$
4: $\quad$ **dacă** $T[j] =$ NIL **atunci**
5: $\quad\quad T[j] \leftarrow k$
6: $\quad\quad$ **returnează** j
7: $\quad$ **altfel**
8: $\quad\quad i \leftarrow i + 1$
9: **până când** $i = m$
10: **eroare** "depăşire tabela de dispersie"

Algoritmul pentru căutarea unei chei k examinează aceeaşi secvenţă de locaţii pe care o foloseşte şi algoritmul de inserare atunci când s-a inserat cheia k. Prin urmare, căutarea se poate termina (fără succes) când se întâlneşte o locaţie liberă, pentru că k ar fi fost inserat în acea locaţie şi nu mai încolo în secvenţa de verificare. (Se observă că această argumentaţie presupune că o dată introduse, cheile nu mai sunt şterse din tabela de dispersie.) Procedura DISPERSIE-CAUTĂ are ca intrare o tabelă de dispersie T şi o cheie k şi returnează j dacă locaţia j conţine cheia k sau NIL dacă cheia k nu există în tabela T.

DISPERSIE-CAUTĂ(T, k)
1: $i \leftarrow 0$
2: **repetă**
3: $\quad j \leftarrow h(k, i)$
4: $\quad$ **dacă** $T[j] = k$ **atunci**
5: $\quad\quad$ **returnează** j
6: $\quad\quad i \leftarrow i + 1$
7: **până când** $T[j] =$ NIL **sau** $i = m$
8: **returnează** NIL

Ştergerea dintr-o tabelă de dispersie cu adresare deschisă este dificilă. Când ştergem o cheie dintr-o locaţie i, nu putem marca pur şi simplu acea locaţie ca fiind liberă memorând în ea

valoarea NIL. Procedând astfel, va fi imposibil să accesăm orice cheie k a cărei inserare a verificat locaţia i şi a găsit-o ocupată. O soluţie este de a marca locaţia, memorând în ea valoarea specială ŞTERS în loc de NIL. Apoi vom modifica procedura DISPERSIE-INSEREAZĂ astfel încât să trateze astfel de locaţii ca şi când ar fi libere, astfel încât o nouă cheie să poată fi inserată. Nu este necesară nici o modificare în DISPERSIE-CAUTĂ, deoarece va trece peste valorile ŞTERS în timpul căutării. Procedând astfel, timpii de căutare nu mai depind de factorul de încărcare α şi din acest motiv în cazul în care cheile trebuie şterse, se preferă, în mod uzual, înlănţuirea ca tehnică de rezolvare a coliziunilor.

În analiza noastră, folosim ipoteza ***dispersiei uniforme:*** presupunem că fiecare cheie considerată poate avea, cu aceeaşi probabilitate, oricare dintre cele $m!$ permutări ale mulţimii $\{0, 1, ..., m-1\}$ ca secvenţă de verificare. Dispersia uniformă generalizează noţiunea de dispersie uniformă simplă, definită anterior, pentru situaţia în care funcţia de dispersie produce nu doar un singur număr, ci o întreagă secvenţă de verificare. Oricum, dispersia uniformă reală este dificil de implementat şi în practică sunt folosite aproximări convenabile (ca dispersia dublă, definită mai târziu).

În mod obişnuit sunt utilizate trei tehnici pentru a calcula secvenţele de verificare necesare adresării deschise: verificarea liniară, verificarea pătratică şi dispersia dublă. Toate aceste tehnici garantează că $\langle h(k,0), h(k,1), ..., h(k,m-1)\rangle$ este o permutare a lui $\langle 0, 1, ..., m-1\rangle$ pentru fiecare cheie k. Nici una din aceste tehnici nu satisface condiţia dispersiei uniforme, pentru că nici una dintre ele nu e capabilă să genereze mai mult de m^2 secvenţe de verificare diferite (în loc de $m!$ cât cere dispersia uniformă). Dispersia dublă are cel mai mare număr de secvenţe de verificare şi, după cum era de aşteptat, dă cele mai bune rezultate.

Verificarea liniară

Fiind dată o funcţie de dispersie ordinară $h' : U \to \{0, 1, ..., m-1\}$, metoda ***verificării liniare*** foloseşte funcţia de dispersie

$$h(k,i) = (h'(k) + i) \mod m$$

pentru $i = 0, 1, ..., m-1$. Fiind dată o cheie k, prima locaţie examinată (verificată) este $T[h'(k)]$. Apoi examinăm locaţia $T[h'(k)+1]$ şi aşa mai departe până la locaţia $T[m-1]$. Apoi continuăm circular cu locaţiile $T[0], T[1], ...$, până când, în final, verificăm locaţia $T[h'(k)-1]$. Deoarece poziţia de start a verificării determină întreaga secvenţă de verificare liniară, prin care sunt folosite doar m secvenţe de verificări distincte.

Verificarea liniară este uşor de implementat, dar apare o problemă cunoscută sub numele de ***grupare primară***. În acest caz se formează şiruri lungi de locaţii ocupate, crescând timpul mediu de căutare. De exemplu, dacă avem $n = m/2$ chei într-o tabelă, în care fiecare locaţie de indice par este ocupată şi fiecare locaţie de indice impar este liberă, atunci căutarea fără succes necesită, 1,5 verificări în cazul mediu. Dacă primele $n = m/2$ locaţii sunt cele ocupate, numărul mediu de verificări creşte la aproximativ $n/4 = m/8$. Grupările au probabilitate mare de apariţie, pentru că dacă o locaţie liberă este precedată de i locaţii ocupate, atunci probabilitatea ca locaţia liberă să fie următoarea completată este $(i+1)/m$, comparativ cu probabilitatea de $1/m$ pentru cazul în care locaţia precedentă ar fi fost liberă. Prin urmare, şirurile de locaţii ocupate tind să se lungească şi verificarea liniară nu este o aproximare foarte bună a dispersiei uniforme.

Verificarea pătratică

Verificarea pătratică foloseşte o funcţie de dispersie de forma

$$h(k,i) = (h'(k) + c_1 i + c_2 i^2) \mod m\,, \tag{12.5}$$

unde (ca şi în cazul verificării liniare) h' este o funcţie de dispersie auxiliară, c_1 şi $c_2 \neq 0$ sunt constante auxiliare şi $i = 0, 1, ..., m-1$. Poziţia verificată iniţial este $T[h'(k)]$; următoarele poziţii examinate sunt decalate cu cantităţi ce depind într-o manieră pătratică de numărul de verificare i. Această metodă lucrează mult mai bine decât verificarea liniară, dar pentru a folosi integral tabela de dispersie valorile lui c_1, c_2 şi m trebuie determinate corespunzător. Problema 12-4 ilustrează o modalitate de determinare a acestor parametri. De asemenea, dacă două chei au aceeaşi poziţie de start a verificării, atunci secvenţele lor de verificare coincid, pentru că $h(k_1, 0 = h(k_2, 0)$ implică $h(k_1, i) = h(k_2, i)$. Această situaţie conduce la o formă mai uşoară de grupare, numită ***grupare secundară***. Ca şi în cazul verificării liniare, verificarea iniţială determină întreaga secvenţă, deci sunt folosite doar m secvenţe de verificare distincte.

Dispersia dublă

dispersia dublă este una dintre cele mai bune metode disponibile pentru adresarea deschisă, deoarece permutările produse au multe din caracteristicile permutărilor alese aleator. ***dispersia dublă*** foloseşte o funcţie de dispersie de forma

$$h(k,i) = (h_1(k) + i h_2(k)) \mod m\,,$$

unde h_1 şi h_2 sunt funcţii de dispersie auxiliare. Poziţia examinată iniţial este $T[h_1(k)]$; poziţiile succesive de verificare sunt decalate faţă de poziţiile anterioare cu $h_2(k)$ modulo m. Astfel, contrar situaţiei de verificarea liniară sau pătratică, în acest caz secvenţa de verificare depinde în două moduri de cheia k, pentru că fie poziţia iniţială de verificare, fie decalajul, fie amândouă pot varia. Figura 12.5 ilustrează un exemplu de inserare prin dispersie dublă.

Valoarea $h_2(k)$ şi dimensiunea tabelei de dispersie m trebuie să fie prime între ele pentru a fi parcursă întreaga tabelă de dispersie. În caz contrar, dacă m şi $h_2(k)$ au un cel mai mare divizor comun $d > 1$ pentru o cheie k, atunci o căutare pentru cheia k va examina doar a $(1/d)$-a parte din tabela de dispersie. (Vezi capitolul 33.) O modalitate convenabilă de a asigura această condiţie este de a avea un număr m ca o putere a lui 2 şi de a construi h_2 astfel încât să producă întotdeauna un număr impar. O altă modalitate este de a avea m prim şi de a construi h_2 astfel încât să returneze întotdeauna un întreg pozitiv mai mic decât m. De exemplu, putem alege m prim şi

$$\begin{aligned} h_1(k) &= k \mod m\,, \\ h_2(k) &= 1 + (k \mod m')\,, \end{aligned}$$

unde m' este ales să fie un pic mai mic decât m (să zicem $m-1$ sau $m-2$). De exemplu, dacă $k = 123456, m = 701$ şi $m' = 700$, avem $h_1(k) = 80$ şi $h_2(k) = 257$, deci prima verificare se află la poziţia 80 şi fiecare a 257-a locaţie (modulo m) este examinată până când cheia este găsită sau a fost examinată toată tabela.

dispersia dublă reprezintă o îmbunătăţire faţă de verificarea liniară sau pătratică în sensul că sunt folosite $\Theta(m^2)$ secvenţe de verificare, faţă de $\Theta(m)$, pentru că fiecare pereche posibilă $(h_1(k), h_2(k))$ produce o secvenţă de verificare distinctă c si când cheia variază, poziţia iniţială

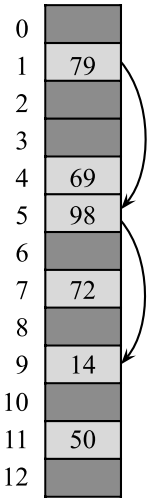

Figura 12.5 Inserarea prin dispersie dublă. Avem o tabelă de dispersie de dimensiune 13, cu $h_1(k) = k \bmod 13$ şi $h_2(k) = 1 + (k \bmod 11)$. Deoarece $14 \equiv 1 \bmod 13$ şi $14 \equiv 3 \bmod 11$, cheia 14 va fi inserată în locaţia liberă 9, după ce locaţiile 1 şi 5 au fost examinate şi găsite ca fiind deja ocupate.

a verificării $h_1(k)$ şi decalajul $h_2(k)$ pot varia independent. Ca rezultat, performanţa dispersiei duble apare ca fiind foarte apropiată de performanţa schemei "ideale" a dispersiei uniforme.

Analiza dispersiei cu adresare deschisă

Analiza noastră pentru adresarea deschisă, este, ca şi analiza pentru înlănţuire, exprimată în termenii factorului de încărcare α al tabelei de dispersie, când n şi m tind spre infinit. Să reamintim că dacă sunt memorate n elemente într-o tabelă cu m locaţii, numărul mediu de elemente pe locaţie este $\alpha = n/m$. Desigur, prin adresare deschisă, avem cel mult un element într-o locaţie şi, prin urmare, $n \leq m$, ceea ce implică $\alpha \leq 1$.

Presupunem că este folosită dispersia uniformă. În această schemă idealizată, secvenţa de verificare $\langle h(k,0), h(k,1), ..., h(k,m-1)\rangle$ pentru orice cheie k poate să apară sub forma oricărei permutări a mulţimii $\langle 0, 1, ..., m-1\rangle$. Aceasta înseamnă că fiecare secvenţă de verificare posibilă poate fi folosită ca secvenţă de verificare pentru o inserare sau o căutare. Desigur, o cheie dată are o secvenţă de verificare unică, fixă, asociată ei; considerând distribuţia de probabilitate pe spaţiul cheilor şi operaţia funcţiei de dispersie pe chei, fiecare secvenţă de verificare posibilă are aceeaşi probabilitate de apariţie.

Analizăm acum numărul mediu de verificări pentru dispersia cu adresare deschisă, în ipoteza dispersiei uniforme şi începem cu o analiză a numărului de verificări ce apar într-o căutare fără succes.

Teorema 12.5 Fiind dată o tabelă de dispersie cu adresare deschisă cu factorul de încărcare $\alpha = n/m < 1$, în ipoteza dispersiei uniforme, numărul mediu de verificări dintr-o căutare fără succes este cel mult $1/(1-\alpha)$.

Demonstraţie. Într-o căutare fără succes, fiecare verificare, cu excepţia ultimei, accesează o

locaţie ocupată care nu conţine cheia dorită, iar ultima locaţie examinată este liberă. Definim

p_i = Pr { exact i verificări accesează locaţiile ocupate }

pentru $i = 0, 1, 2, ...$ Pentru $i > n$, avem $p_i = 0$, pentru că putem găsi cel mult n locaţii deja ocupate. Astfel, numărul mediu de verificări este

$$1 + \sum_{i=1}^{\infty} ip_i \tag{12.6}$$

Pentru a evalua relaţia (12.6), definim

q_i = Pr { cel puţin i verificări accesează locaţiile ocupate }

pentru $i = 0, 1, 2, ...$ Apoi putem folosi identitatea (6.28):

$$\sum_{i=1}^{\infty} ip_i = \sum_{i=1}^{\infty} q_i \, .$$

Care este valoarea lui q_i pentru $i \geq 1$? Probabilitatea ca prima verificare să acceseze o locaţie ocupată este n/m; prin urmare,

$$q_1 = \frac{n}{m} \, .$$

Prin dispersie uniformă, a doua verificare, în cazul în care este necesară, va fi la una din cele $m-1$ locaţii rămase neverificate, din care $n-1$ sunt ocupate. Facem a doua verificare doar dacă prima verificare accesează o locaţie ocupată; prin urmare,

$$q_2 = \left(\frac{n}{m}\right)\left(\frac{n-1}{m-1}\right) \, .$$

În general, a i-a verificare este necesară doar dacă primele $i-1$ verificări accesează locaţii ocupate şi locaţia examinată este cu aceeaşi probabilitate oricare dintre cele $m-i+1$ locaţii rămase, din care $n-i+1$ sunt ocupate. Deci,

$$q_i = \left(\frac{n}{m}\right)\left(\frac{n-1}{m-1}\right) ... \left(\frac{n-i+1}{m-i+1}\right) \leq \left(\frac{n}{m}\right)^i = \alpha^i$$

pentru $i = 1, 2, ..., n$, pentru că $(n-j)/(m-j) \leq n/m$ dacă $n \leq m$ şi $j \geq 0$. După n verificări, toate cele n locaţii ocupate au fost vizitate şi nu vor fi verificate din nou şi astfel $q_i = 0$ pentru $i > n$.

Acum se poate evalua relaţia (12.6). Presupunând $\alpha < 1$, numărul mediu de verificări într-o căutare fără succes este

$$1 + \sum_{i=0}^{\infty} ip_i = 1 + \sum_{i=1}^{\infty} q_i \leq 1 + \alpha + \alpha^2 + \alpha^3 + ... = \frac{1}{1-\alpha} \, . \tag{12.7}$$

Relaţia (12.7) are o interpretare intuitivă: prima verificare se face întotdeauna, a doua verificare este necesară cu o probabilitate aproximativ egală cu α, a treia verificare este necesară cu o probabilitate aproximativ egală cu α^2 şi aşa mai departe. ∎

Dacă α este o constantă, teorema 12.5 prognozează faptul că o căutare fără succes necesită un timp $O(1)$. De exemplu, dacă tabela de dispersie este pe jumătate plină, numărul mediu de verificări într-o căutare fără succes este cel mult $1/(1-.5) = 2$. Dacă este 90% ocupată, numărul mediu de verificări este cel mult $1/(1-.9) = 10$.

Teorema 12.5 ne dă, aproape imediat, performanţa procedurii DISPERSIE-INSEREAZĂ.

Corolarul 12.6 Inserarea unui element într-o tabelă de dispersie cu adresare deschisă, cu factorul de încărcare α necesită , în cazul mediu, cel mult $1/(1-\alpha)$ verificări, în ipoteza dispersiei uniforme.

Demonstraţie. Un element este inserat doar dacă există spaţiu în tabelă şi, deci, $\alpha < 1$. Inserarea unei chei necesită o căutare fără succes, urmată de o plasare a cheii în prima locaţie găsită liberă. Prin urmare, numărul mediu de verificări este cel mult $1/(1-\alpha)$. ∎

Calculul numărului mediu de verificări pentru o căutare cu succes necesită un pic mai mult de lucru.

Teorema 12.7 Fiind dată o tabelă de dispersie cu adresare deschisă, cu factorul de încărcare $\alpha < 1$, numărul mediu de verificări într-o căutare cu succes este cel mult

$$\frac{1}{\alpha} \ln \frac{1}{1-\alpha} ,$$

în ipoteza dispersiei uniforme şi presupunând că fiecare cheie din tabelă este căutată cu aceeaşi probabilitate.

Demonstraţie. O căutare pentru o cheie k urmează aceeaşi secvenţă de verificare ce a fost urmată când elementul având cheia k a fost inserat. Conform corolarului 12.6, dacă k a fost a $i+1$-a cheie inserată în tabela de dispersie, numărul mediu de verificări făcute într-o căutare a lui k este cel mult $1/(1-i/m) = m(m-i)$. Calculând media peste toate cele n chei din tabela de dispersie, obţinem numărul mediu de verificări dintr-o căutare cu succes:

$$\frac{1}{n} \sum_{i=0}^{n-1} \frac{m}{m-i} = \frac{m}{n} \sum_{i=0}^{n-1} \frac{1}{m-i} = \frac{1}{\alpha}(H_m - H_{m-n}) ,$$

unde $H_i = \sum_{j=1}^{i} 1/j$ este al i-lea număr armonic (definit ca în relaţia (3.5)). Folosind marginile $\ln i \le H_i \le \ln i + 1$ din relaţiile (3.11) şi (3.12), obţinem

$$\begin{aligned} \frac{1}{\alpha}(H_m - H_{m-n}) &= \frac{1}{\alpha} \sum_{k=m-n+1}^{m} 1/k \le \frac{1}{\alpha} \int_{m-n}^{m} (1/x)dx \\ &= \frac{1}{\alpha} \ln(m/(m-n)) = \frac{1}{\alpha} \ln(1/(1-\alpha)) \end{aligned}$$

ca mărginire pentru numărul mediu de verificări într-o căutare cu succes. ∎

Dacă tabela de dispersie este pe jumătate ocupată, numărul mediu de verificări într-o căutare cu succes este mai mic decât 1,387. Dacă tabela de dispersie este 90% ocupată, numărul mediu de verificări este mai mic decât 2,559.

Exerciţii

12.4-1 Se consideră că se inserează cheile 10, 22, 31, 4, 15, 28, 17, 88, 59 într-o tabelă de dispersie de lungime $m = 11$ folosind adresarea deschisă cu funcţia primară de dispersie $h'(k) = k \bmod m$. Ilustraţi rezultatul inserării acestor chei folosind verificarea liniară, verificarea pătratică cu $c_1 = 1$ şi $c_2 = 3$ şi folosind dispersia dublă cu $h_2(k) = 1 + (k \bmod (m-1))$.

12.4-2 Scrieţi un algoritm în pseudocod pentru DISPERSIE-ŞTERGE după descrierea din text şi modificaţi DISPERSIE-INSEREAZĂ şi DISPERSIE-CAUTĂ pentru a încorpora valoarea specială ŞTERS.

12.4-3 ⋆ Să presupunem că folosim dispersia dublă pentru a rezolva coliziunile; mai exact, folosim funcţia de dispersie $h(k,i) = (h_1(k) + ih_2(k)) \bmod m$. Arătaţi că secvenţa de verificare $\langle h(k,0), h(k,1), ..., h(k,m-1)\rangle$ este o permutare a secvenţei de locaţii $\langle 0, 1, ..., m-1\rangle$ dacă şi numai dacă $h_2(k)$ este relativ prim cu m. (*Indicaţie:* Vezi capitolul 33.)

12.4-4 Se consideră o tabelă de dispersie cu adresare deschisă şi cu dispersie uniformă şi factorul de încărcare $\alpha = 1/2$. Daţi margini superioare pentru numărul mediu de verificări într-o căutare fără succes şi pentru numărul mediu de verificări într-o căutare cu succes. Care este numărul mediu de verificări într-o căutare cu succes? Repetaţi calculele pentru factorii de încărcare $3/4$ şi $7/8$.

12.4-5 ⋆ Să presupunem că inserăm n chei într-o tabelă de dispersie de dimensiune m folosind adresarea deschisă şi dispersia uniformă. Notăm $cup(n,m)$ probabilitatea de a nu apare nici o coliziune. Arătaţi că $p(n,m) \leq e^{-n(n-1)/2m}$. (*Indicaţie:* Vezi relaţia (2.7).) Argumentaţi că atunci când n depăşeşte $\sqrt{m}$, probabilitatea evitării coliziunilor descreşte rapid către zero.

12.4-6 ⋆ Se consideră o tabelă de dispersie cu adresare deschisă având factorul de încărcare α. Determinaţi acea valoare nenulă a lui α pentru care numărul mediu de verificări într-o căutare fără succes este de două ori mai mare decât numărul mediu de verificări dintr-o căutare cu succes. Folosiţi estimarea $(1/\alpha)ln(1/(1-\alpha))$ pentru numărul de verificări necesare într-o căutare cu succes.

Probleme

12-1 Marginea celei mai lungi verificări pentru dispersie
O tabelă de dispersie de dimensiune m este folosită pentru a memora n obiecte, cu $n \leq m/2$. Pentru rezolvarea coliziunilor se foloseşte adresarea deschisă.

- ***a.*** În ipoteza dispersiei uniforme, arătaţi că pentru $i = 1, 2, ..., n$, probabilitatea ca a i-a inserare să necesite strict mai mult de k verificări este cel mult 2^{-k}.
- ***b.*** Arătaţi că pentru $i = 1, 2, ..., n$, probabilitatea ca a i-a inserare să necesite mai mult de $2 \lg n$ verificări este cel mult $1/n^2$.

Fie variabila aleatoare X_i, semnificând numărul de verificări necesare pentru a i-a inserare. Aţi arătat la punctul (b) că $\Pr\{X_i > 2 \lg n\} \leq 1/n^2$. Fie variabila aleatoare $X = \max_{1 \leq i \leq n} X_i$ care semnifică numărul maxim de verificări necesare oricăreia din cele n inserări.

c. Arătaţi că $\Pr\{X > 2\lg n\} \leq 1/n$.

d. Arătaţi că lungimea medie a celei mai lungi secvenţe de verificare este $\mathrm{E}[X] = O(\lg n)$.

12-2 Căutarea într-o mulţime statică

Se cere să se implementeze o mulţime dinamică având n elemente, în care cheile sunt numere. Mulţimea este statică (nu există operaţiile INSEREAZĂ şi ŞTERGE) şi singura operaţie necesară este CAUTĂ. Se pune la dispoziţie un timp arbitrar pentru a preprocesa cele n elemente, astfel încât operaţiile de căutare să se execute rapid.

a. Arătaţi că procedura CAUTĂ poate fi implementată astfel încât, în cazul cel mai defavorabil să se execute într-un timp $O(\lg n)$, fără a folosi spaţiu suplimentar faţă de cel necesar memorării elementelor.

b. Se consideră implementarea mulţimii printr-o tabelă de dispersie cu adresare deschisă având m locaţii care foloseşte dispersia uniformă. Care este spaţiul minim necesar (de lungime $m-n$) suplimentar pentru ca performanţa, în cazul mediu a unei căutări fără succes, să fie cel puţin la fel de bună ca marginea de la punctul (a)? Răspunsul trebuie să fie o mărginire asimptotică a lui $m - n$ în funcţie de n.

12-3 Margini pentru mărimea locaţiei la înlănţuire

Să presupunem că avem o tabelă de dispersie cu n locaţii, având coliziuni rezolvate prin înlănţuire şi să presupunem că se inserează n chei în tabelă. Fiecare cheie se poate dispersa, cu aceeaşi probabilitate, în oricare dintre locaţii. Fie M numărul maxim de chei într-o locaţie, după ce toate cheile au fost inserate. Demonstraţi că $\mathrm{E}[M]$, valoarea medie a lui M, are o margine superioară egală cu $O(\lg n/\lg\lg n)$.

a. Fie Q_k probabilitatea ca un număr de k chei să se disperseze într-o locaţie particulară. Argumentaţi că probabilitatea Q_k este dată de

$$Q_k = \left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{n-k} \binom{n}{k}.$$

b. Fie P_k probabilitatea ca $M = k$, adică probabilitatea ca locaţia care conţine cele mai multe chei să conţină k chei. Arătaţi că $P_k \leq nQ_k$.

c. Folosiţi aproximarea Stirling, din relaţia (2.11), pentru a arăta că $Q_k < e^k/k^k$.

d. Arătaţi că există o constantă $c > 1$ astfel încât $Q_{k_0} < 1/n^3$ pentru $k_0 = c\lg n/\lg\lg n$. Deduceţi că $P_k < 1/n^2$ pentru $k = c\lg n/\lg\lg n$.

e. Arătaţi că

$$\mathrm{E}[M] \leq \Pr\left\{M > \frac{c\lg n}{\lg\lg n}\right\} \cdot n + \Pr\left\{M \leq \frac{c\lg n}{\lg\lg n}\right\} \cdot \frac{c\lg n}{\lg\lg n}.$$

Deduceţi că $E[M] = O(\lg n/\lg\lg n)$.

12-4 Verificarea pătratică
Să presupunem că avem o cheie k, care trebuie căutată într-o tabelă de dispersie având poziţiile $0, 1, ..., m-1$ şi să presupunem că avem o funcţie de dispersie h care transformă spaţiul cheilor în mulţimea $\{0, 1, ..., m-1\}$. Schema de căutare este următoarea.

1. Se calculează valoarea $i \leftarrow h(k)$ şi se iniţializează $j \leftarrow 0$.
2. Se verifică poziţia i pentru cheia dorită k. Dacă se găseşte sau dacă poziţia este liberă atunci căutarea se termină.
3. Se fac atribuirile $j \leftarrow (i + j) \bmod m$ şi se revine la pasul 2.

Se presupune că m este o putere a lui 2.

a. Arătaţi că această schemă este o instanţiere a schemei generale de "verificare pătratică" evidenţiind valorile specifice ale constantelor c_1 şi c_2 din relaţia (12.5).

b. Demonstraţi că acest algoritm examinează fiecare poziţie din tabelă în cazul cel mai defavorabil.

12-5 dispersie k-universală
Fie $\mathcal{H} = \{h\}$ o clasă de funcţii de dispersie în care fiecare h transformă universul U al cheilor în $\{0, 1, ..., m-1\}$. Spunem că $\mathcal{H}$ este ***k-universală*** dacă pentru fiecare secvenţă fixă de k chei distincte $\langle x_1, x_2, ..., x_k \rangle$ şi pentru fiecare funcţie h aleasă aleator din $\mathcal{H}$, secvenţa $\langle h(x_1), h(x_2), ..., h(x_k) \rangle$ este, cu aceeaşi probabilitate, oricare dintre cele m^k secvenţe de lungime k având elemente din $\{0, 1, ..., m-1\}$.

a. Arătaţi că dacă $\mathcal{H}$ este 2-universală atunci ea este universală.

b. Arătaţi că clasa $\mathcal{H}$ definită în secţiunea 12.3.3 nu este 2-universală.

c. Arătaţi că dacă modificăm definiţia lui $\mathcal{H}$ din secţiunea 12.3.3 astfel încât fiecare funcţie să conţină un termen constant b, adică dacă înlocuim $h(x)$ cu

$$h_{a,b}(x) = \sum_{i=0}^{r} a_i x_i + b \bmod m \,,$$

atunci $\mathcal{H}$ este 2-universală.

Note bibliografice

Knuth [123] şi Gonnet [90] sunt referinţe excelente pentru algoritmi de dispersie. Knuth îl creditează pe H. P. Luhn (1953) cu inventarea tabelelor de dispersie, împreună cu metoda înlănţuirii pentru rezolvarea coliziunilor. Aproximativ în aceeaşi perioadă, G. M. Amdahl a avut ideea originală a adresării deschise.