UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI 1818

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Proiectarea Algoritmilor

32. Algoritmi de tip Las Vegas

Bibliografie

[1] C. Giumale – Introducere in Analiza Algoritmilor – cap. 6.1

[2] Cormen – Introducere in algoritmi – cap. 8.3

[3] http://www.soe.ucsc.edu/classes/cmps102/Spring04/TantaloAsymp.pdf

[4] http://www.mersenne.org/

Algoritmi Las Vegas

- Găsesc soluția corectă a problemei, însă timpul de rezolvare nu poate fi determinat cu exactitate.
- Creșterea timpului de rezolvare → creșterea probabilității de terminare a algoritmului.
- Timp = ∞ → algoritmul se termina sigur (soluția e optimă).
- Probabilitatea de găsire a soluției creste extrem de repede astfel încât să se determine soluția corectă într-un timp suficient de scurt.

Complexitatea algoritmilor Las Vegas

- Definiția 6.1: Algoritmii Las Vegas au complexitatea $f(n) = O(g(n))$ dacă $\exists\ c > 0$ și $n_0 > 0$ a.î. $\forall\ n \geq n_0$ avem $0 < f(n) < c\ \alpha\ g(n)$ cu o probabilitate de cel puțin $1 - n^{-\alpha}$ pentru $\alpha > 0$ fixat și suficient de mare.

Exemplu algoritm Las Vegas

- Problemă:
  - Capitolele unei cărți sunt stocate într-un fișier text sub forma unei secvențe nevide de linii;
  - Fiecare secvență este precedată de o linie contor ce indică numărul de linii din secvență;
  - Fiecare linie din fișier este terminată prin CR, LF;
  - Toate liniile din secvență au aceeași lungime;
  - Fiecare secvență de linii conține o linie (titlul capitolului) ce se repetă și care apare în cel puțin 10% din numărul de linii al secvenței.
  - Secvențele sunt lungi.☺
- Cerință:
  - Pentru fiecare secvență de linii să se tipărească titlul capitolului (linia care se repetă).

Rezolvare "clasică"

Detectează_linii(fișier)

- **Pentru fiecare** Secv ∈ fișier
  - **Pentru** i **de la** 0 **la** dim(Secv)
    - **Pentru** j **de la** i + 1 **la** dim(Secv)
      - **Dacă** linie(i,Secv) = linie(j,Secv) **atunci**
        - **Întoarce** (linie(i,Secv);

prelucrare secvență

Complexitate – O(dim(Secv)^2)

Algoritm Las Vegas pentru rezolvarea problemei

- Secțiunea “prelucrare secvență” se înlocuiește cu următoarea funcție:

- Selecție_linii(n,secv) // n = dim secv
  - **Cât timp**(1) // mereu
    - i = random(0,n-1) // selectez o linie
    - j = random(0,n-1) // și încă una
    - **Dacă** i != j **și** linie(i,Secv) = linie(j,Secv) **atunci** // le compar
      - **Întoarce** linie(i,Secv) // am găsit linia

Analiza algoritmului Las Vegas (I)

- Notații:
  - n = numărul de linii din secvența curentă;
  - q = ponderea liniei repetate în secvență;
  - r = numărul de apariții al liniei repetate: r = n * q / 100;
  - m = numărul de pași necesari terminării algoritmului;
  - P_k = probabilitatea ca la pasul k să fie satisfăcută condiția de terminare a algoritmului;
  - P(m) = probabilitatea ca algoritmul să se termine după m pași.

Analiza algoritmului Las Vegas (II)

- Probabilitatea ca la pasul k linia i să fie una din liniile repetate este r / n.
- Probabilitatea ca la pasul k linia j să fie una din liniile repetate (diferită de i) este (r - 1) / n.
- Condiția de terminare: cele 2 evenimente trebuie să se producă simultan:

$$P_k = r / n * (r-1) / n = q / 100 * (q / 100 – 1 / n)$$

Analiza algoritmului Las Vegas (III)

- Probabilitatea ca algoritmul să NU se termine după m pași:
  - $\Pi_{k = 1 \rightarrow m}(1 - P_k) = \Pi_{k = 1 \rightarrow m} [1 – q / 100 * (q / 100 – 1 / n)] = [1 – q / 100 * (q / 100 – 1 / n)]^m$
- → $P(m) = 1 - [1 – q / 100 * (q / 100 – 1 / n)]^m$
- Pp: n > 100; q > 10%
- → $P(m) \geq 1 – [1 – q * (q – 1) / 10.000]^m$

Comparație timp de rulare

- q = 10%:
  - 3500 pași P = 1;
  - 1000 pași – P = 0,9988.
- q = 20%:
  - 1000 pași P = 1.
- q = 30%:
  - 500 pași P = 1.
- Varianta clasică: cazul cel mai defavorabil – 10000 pași.

Complexitate algoritm Las Vegas

- Algoritmii Las Vegas au complexitatea $f(n) = O(g(n))$ dacă $\exists\, c > 0$ si $n_0 > 0$ a.i. $\forall\, n \geq n_0$ avem $0 < f(n) < c\,\alpha\, g(n)$ cu o probabilitate de cel puțin $1 - n^{-\alpha}$ pentru $\alpha > 0$ fixat si suficient de mare.

- Arătăm că $f(n) = O(\lg(n))$:
  - Notăm: $a = 1 – q * (q - 1) / 10.000$;
  - $1 – P(c\,\alpha \lg(n))$ = probabilitatea ca algoritmul să nu se termine în $c\,\alpha \lg(n)$ pași;
  - $P(c\,\alpha \lg(n)) \geq 1- a^{c\,\alpha \lg(n)} \rightarrow 1 – P(c\,\alpha \lg(n)) \leq a^{c\,\alpha \lg(n)} = n^{c\,\alpha \lg(a)} = n^{-c\,\alpha \lg(1/a)}$ pentru că $0 < a < 1$;
  - Dacă alegem $c \geq \lg^{-1}(1/a) \rightarrow 1 – P(c\,\alpha \lg(n)) \leq n^{-\alpha} \rightarrow P(c\,\alpha \lg(n)) \geq 1 - n^{-\alpha} \rightarrow$ algoritmul se termină în $\lg^{-1}(1/a) \lg(n)$ pași cu o probabilitate $\geq 1 - n^{-\alpha} \rightarrow$ (definiție) $f(n) = O(\lg(n))$.

PA