DISTRIBUIRANI ALGORITMI I SISTEMI

Iz kursa CSCE 668
Proleće 2014

Autor izvorne prezentacije:
Prof. Jennifer Welch

Izbor lidera u sinhronim prstenima

- Ovo je jedan prost algoritam; predpost. da su id-ovi nenegativni celi brojevi
- Grupišimo runde u **faze**, svaka faza ima n rundi; počnimo brojanje od 0
- U fazi i, procesor sa id i, ako on postoji, šalje poruku oko prstena i biva izabran

Primer prostog sinhronog algoritma

- $n = 4$, najmanji id je 7
- U fazama 0 do 6 (i odgovarajućim rundama 1 do 28), ne šalje se ni jedna poruka
- Na početku faze 7 (runda 29), proc. sa id 7 šalje poruku koja se prosleđuje oko prstena

Primer prostog sinhronog algoritma

- $n = 4$, najmanji id je 7
- U fazama 0 do 6 (i odgovarajućim rundama 1 do 28), ne šalje se ni jedna poruka
- Na početku faze 7 (runda 29), proc. sa id 7 šalje poruku koja se prosleđuje oko prstena

na osnovu sinhronizma i poznavanja n

Analiza prostog algoritma

- **Korektnost:** Lako se vidi
- **Broj poruka:** O(*n*), što je optimalno
- **Vreme izvršenja:** O(*n*m*), gde je *m* najmanji id u prstenu
  - **nije ograničeno ni jednom funkcijom od n, što je nepoželjno**

Drugi sinhroni algoritam

- Radi u malo slabijem modelu od predhodnog sinhronog algoritma:
  - procesori ne moraju svi krenuti u istoj rundi; procesor se ili spontano probudi ili prvo dobije poruku
  - uniforman (ne oslanja se na poznavanje *n*)

Drugi sinhroni algoritam

- Procesor koji se spontano probudi je **aktivan**; šalje svoj id u **brzim** porukama (1 luk po rundi)
- Procesor koji se probudi kad primi poruku je **relej**; nikad nije konkurencija
- Brza poruka koja nosi id *m* postaje **spora** ako dođe do aktivnog procesora; dalje putuje brzinom 1 luk po 2^m rundi
- Procesor prosleđuje samo poruku čiji id je manji od bilo kog id koji je ranije video
- Ako procesor primi svoj sopstveni id, izabira sebe

Analiza sinhronih algoritama

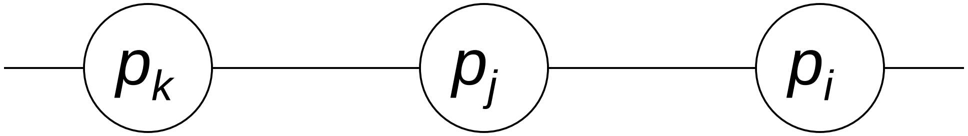
- **Korektnost:** ubedite sebe da se izabira aktivni procesor sa najmanjim id
- **Broj poruka:** Pobednikova poruka je najbrža. Dok ona prelazi preko prstena, druge por. su sporije, pa bivaju prestignute i zaustavjene pre nego što je previše poruka poslato
  - Kvantifikujmo sada ove ideje…

Broj poruka

- Podelimo poruke u tri vrste:
  1. brze por.
  2. spore por. poslate dok je liderova por. brza
  3. spore por. poslate dok je liderova por. spora
- Dalje, prebrojmo poruke svake vrste

Broj poruka tipa 1

- Pokažimo da ni jedan procesor ne prosleđuje više od jedne brze por.:



- Predpost. da p_i prosleđuje brze por. od p_j i p_k . Kada brza por. od p_k stigne do p_j :
  - ili je p_j već poslao svoju brzu por., pa por. od p_k postaje spora pre nego što stigne do p_i, ili
  - p_j još nije poslao svoju brzu por., ali je sad neće ni poslati
- Zato poruka tipa 1 ima najviše n

Broj poruka tipa 2

(spore poslate dok je liderova por. brza)

- Liderova por. je brza u toku najviše n rundi
  - do tada bi se ona vratila do lidera
- Spora por. i se prosleđuje $n/2^i$ puta u n rundi
- Broj por. je maksimalan kada su id-ovi najmanji mogući (0 do $n-1$ i lider je 0)
- Poruka tipa 2 ima najviše:

$$\sum_{i=1}^{n-1} (n/2^i) \leq n$$

Broj poruka tipa 3

(spore por. poslate dok je liderova spora)

- Max broj rundi tokom kojih je liderova poruka spora je $n*2^L$ (L je liderov id)
- Nema slanja por. nakon što se liderova por. vratila lideru
- Spora por. i se prosleđuje $n*2^L/2^i$ puta tokom $n*2^L$ rundi
- Najgori slučaj je kada su id-ovi između L i $L+n-1$
- Poruka tipa 3 ima najviše:

$$\sum_{i=L}^{L+n-1}(n*2^L/2^i) \leq 2n$$

Ukupan broj poruka

- Pokazali smo
  - broj poruka tipa 1 je najviše n
  - broj poruka tipa 2 je najviše n
  - broj poruka tipa 3 je najviše $2n$
- Zato je ukupan broj poruka najviše $4n = O(n)$

Vreme izvršenja sinhronih algoritama

- Vreme izvršenja je O($n\ 2^x$), gde je x najmanji id
- Čak gore od predhodnog algoritma, gde je O($n\ x$)
- Oba algoritma imaju 2 potencijalno neželjene osobine:
  - oslanjaju se na brojne vrednosti id-ova kod brojanja
  - broj rundi zavisi od min id, koji može da nema veze sa n
- Može se dokazati da ako br. por. treba da bude linearna f-ija od n, onda algoritam mora da se oslanja na brojne vrednosti id-ova

Nastavak o anonimnim prstenima

- Izbor lidera nije moguć u anonimnim prstenima
- Nema načina da se razbije simetrija
- Nema *determinističnog* algoritma koji radi u *svakom* izvršenju
- Drugi način da se razbije simetrija, koji radi u *nekim* (ali ne svim) izvršenjima je da se koristi **randomizacija**

Randomiziran algoritam

- U svakom koraku računanja, procesor prima slučajan broj kao dodatni ulazi pod. u svojoj funkciji za promenu stanja

Nastavak defincije LE problema

- Oslabljena defincija problema u odnosu na original:
- Najviše 1 lider se izabira u svakom stanju svakog prihvatljivog izvršenja
  - Isto kao predhodna definicija
- Bar 1 lider se izabira „sa velikom verovatnoćom“
  - slabije od predhodne definicije
- Ali šta zanči „sa velikom verovatnoćom“?

Randomiziran LE algoritam

- Predpostavimo sinhroni model
- Na početku:
  - postavi *id* na 1 sa verovatnoćom $1 - 1/n$ i na 2 sa verovatnoćom $1/n$
  - pošalji *id* u levo
- Kada je poruka *M* primljena:
  - ako *M* sadrži *n* id-ova onda
    - ako je *id* jedinstven maksimum u *M* onda izaberi sebe
    - inače nije izabran
  - inače dodaj *id* u *M* i pošalji u levo

Analiza randomiziranog LE alg.

- Koristi $O(n^2)$ poruka
- Nema nikada više od jednog lidera
- Ponekad nema lidera
  - lider se izabira samo ako postoji tačno jedan procesor koji postavi svoj id na 2
- Koliko često nema lidera, tj. kolika je verovatnoća?
- Potrebne su neke dodatne defincije…

Slučajni izbori i verovatnoće

- Pošto je sistem sinhron, prihvatljivo izvršenje algoritma je određeno isključivo početnim slučajnim izborima
- Nazovimo ovu kolekciju *slučajnim izborima*

$$RC = \langle r_0, r_1, \ldots, r_{n-1} \rangle$$

  gde je svaki r_i ili 1 ili 2
- Neka je exec(*RC*) izvršenje koje proističe iz *RC*
- **Definicija:** Za bilo koji predikat *P* nad izvršenjima, *Pr*[*P*] je verovatnoća od

$$\{RC : exec(RC) \text{ zadovoljava } P\}$$

  tj., proporicija slučajnih izbora iz kojih proističu izvršenja koja zadovoljavaju *P*

Verovatnoća izbora lidera

- Neka je P predikat „postoji lider“.
- $Pr[P]$ = verovatnoća da RC sadrži tačno jednu 2-ku

$$= \binom{n}{1} \cdot \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1} = \left(1 - \frac{1}{n}\right)^{n-1} \approx \frac{1}{e} \approx .37$$

Poboljšanje verovatnoće izbora lidera

- Ako proc. primete da nema lidera, mogu da probaju ponovo
- Svaka iteracija osnovnog algoritma je **faza**.
- Nastavlja se sa pokušajima do konačnog uspeha
- Slučajni izbori koji definišu izvršenje se sastoje od beskonačne sekvence 1-ca i 2-ki, po jedne za svaki proc.
- Moguće je da se algoritam ne završi

Verovatnoća da se alg. ne završi

- Verovatnoća završetka u jednoj fazi je $(1 - 1/n)^{n-1}$
- Verovatnoća da nema završetka u jednoj fazi je $1 - (1 - 1/n)^{n-1}$
- Verovatnoća da nema završetka u k faza je $(1 - (1 - 1/n)^{n-1})^k$ pošto su faze nezavisne
- Zadnji izraz teži ka 0 kako se k povećava

Očekivani broj faza

- **Definicija:** Očekivana vrednost slučajne promenljive T je

$$E[T] = \sum_k k \, Pr[T = k]$$

- Neka je T broj faza do završetka
- $Pr[T = k]$

  $= Pr[\text{prvih } k-1 \text{ faza neuspešno \& } k\text{-ta uspešna}]$

  $= (1 - (1 - 1/n)^{n-1})^{k-1} (1 - 1/n)^{n-1}$

  $= (1 - p)^{k-1} p$, gde je $p = (1 - 1/n)^{n-1}$
- Ovo je geometrijska slučajna promenljiva sa očekivanom vrednošću $p^{-1} < e$.
- Sledi da je očekivani broj faza je < 3