Implementarea unei baze de date distribuite cu suport pentru notificări în timp real

Conducător ştiinţific
S.l dr. ing. Vlad-Alexandru GROSU

Absolvent
Ion STOICAN

2017

Cuprins

- Obiectivele lucrării
- Cui se adresează ?
- Avantajele oferite de protocolul gossip
- De ce limbajul GO ?
- Aplicația server
- Interfața grafică de monitorizare
- Concluzii și îmbunătățiri viitoare
- Contribuții personale

Obiectivele lucrării

Obiectivele lucrării

1. Studierea arhitecturii unei baze de date distribuite de tip nosql (cheie - valoare) folosind protocolul gossip, urmărind:
    - scalabilitatea (eng. scalability)
    - disponibilitatea aplicației (eng. availability)
2. Implementarea
3. Testarea performanțelor

Cui se adresează ?

Aplicațiilor care au nevoie:

- să stocheze un volum mare de date
- ca baza de date să răspundă în orice situație
- notificări în timp real

Compromisuri acceptabile:

- accesarea datelor doar după o cheie
- datele pot fi pierdute

Avantajele oferite de protocolul gossip

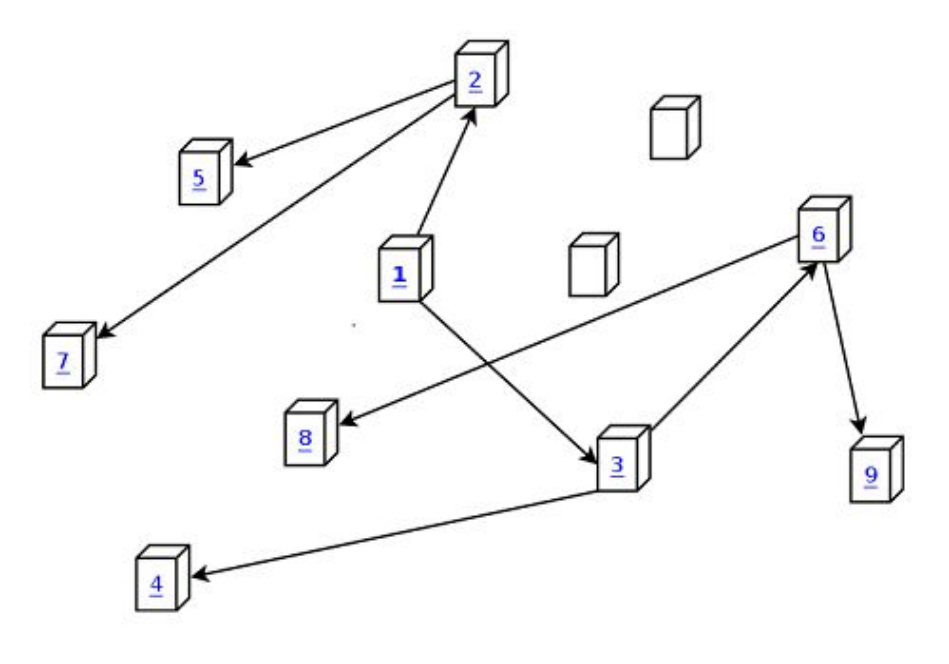
- Scalabil liniar cu numărul de noduri
- Toate nodurile sunt egale din punct de vedere al funcționalității
- Nu este nevoie de sincronizare globală

De ce limbajul GO ?

- “goroutines” - fire de execuție paralelă ce consumă mult mai puțină memorie în comparație cu thread-urile (2kB vs. 1MB)
- Se compilează direct în cod mașină - simplu de executat
- Performantă oferită
- Garbage collected - automatizează managementul memoriei alocate

Aplicația server

Arhitectură

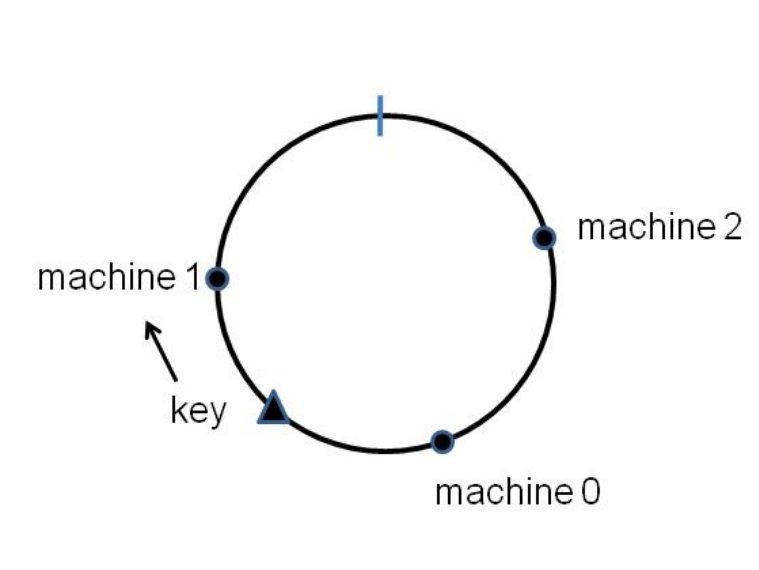


- Distribuită, de tip gossip(peer-to-peer)
- Client-server(http)

Operații posibile

1. Citirea valorii asociate unei chei
2. Scrierea valorii asociate unei chei
3. Notificarea operațiunilor ce se fac asupra unei chei

Algoritmul de partiționare a datelor



Consistent hashing

Două implementări ale funcției de hashing:

- md5
- crc32

Modulul de stocare a datelor

Două implementări:

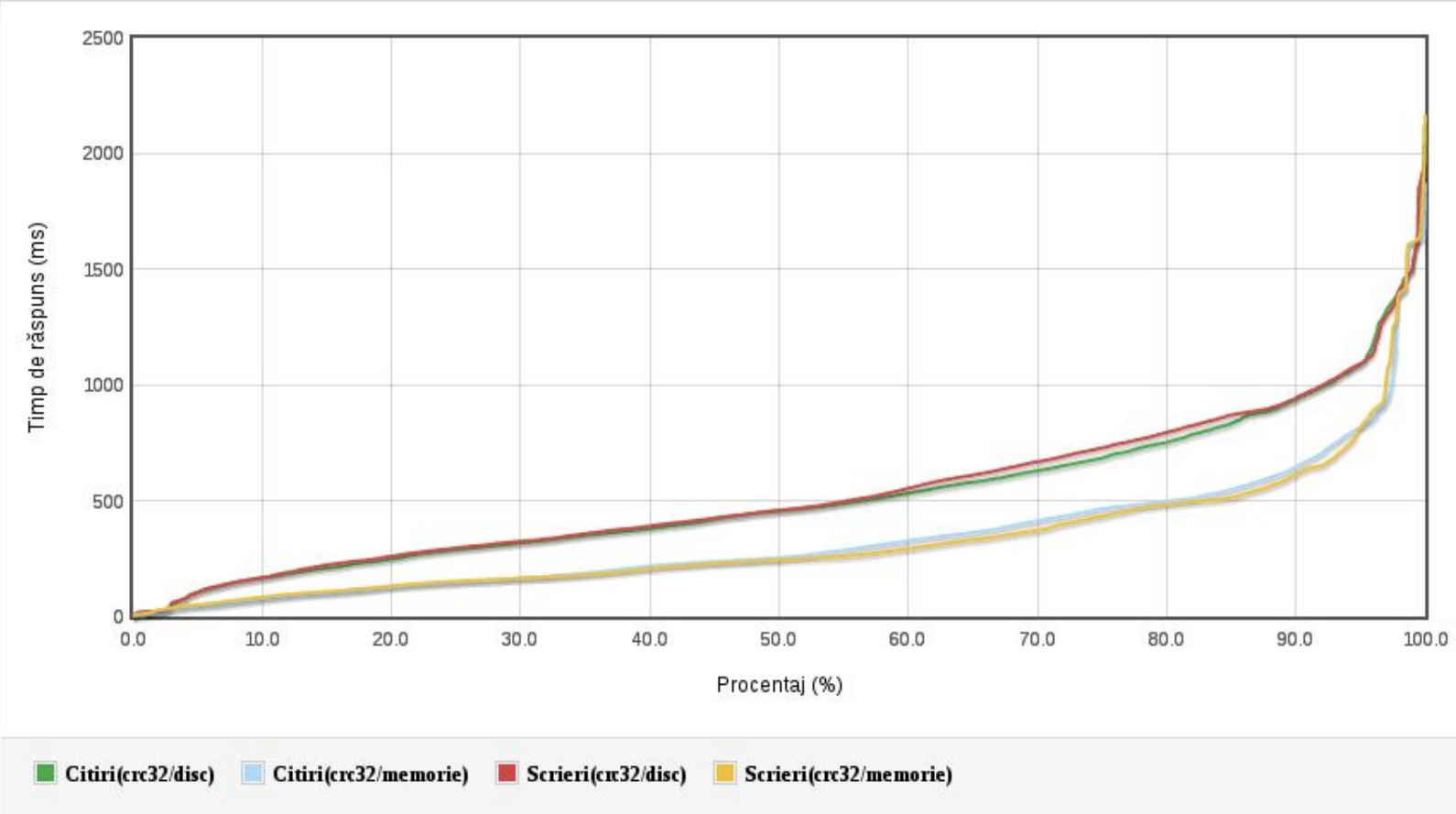
- memorie - pentru a evita a face din modulul de stocare un punct de gâtuire
- disc - permite simularea în condiții reale

Evaluarea performanțelor

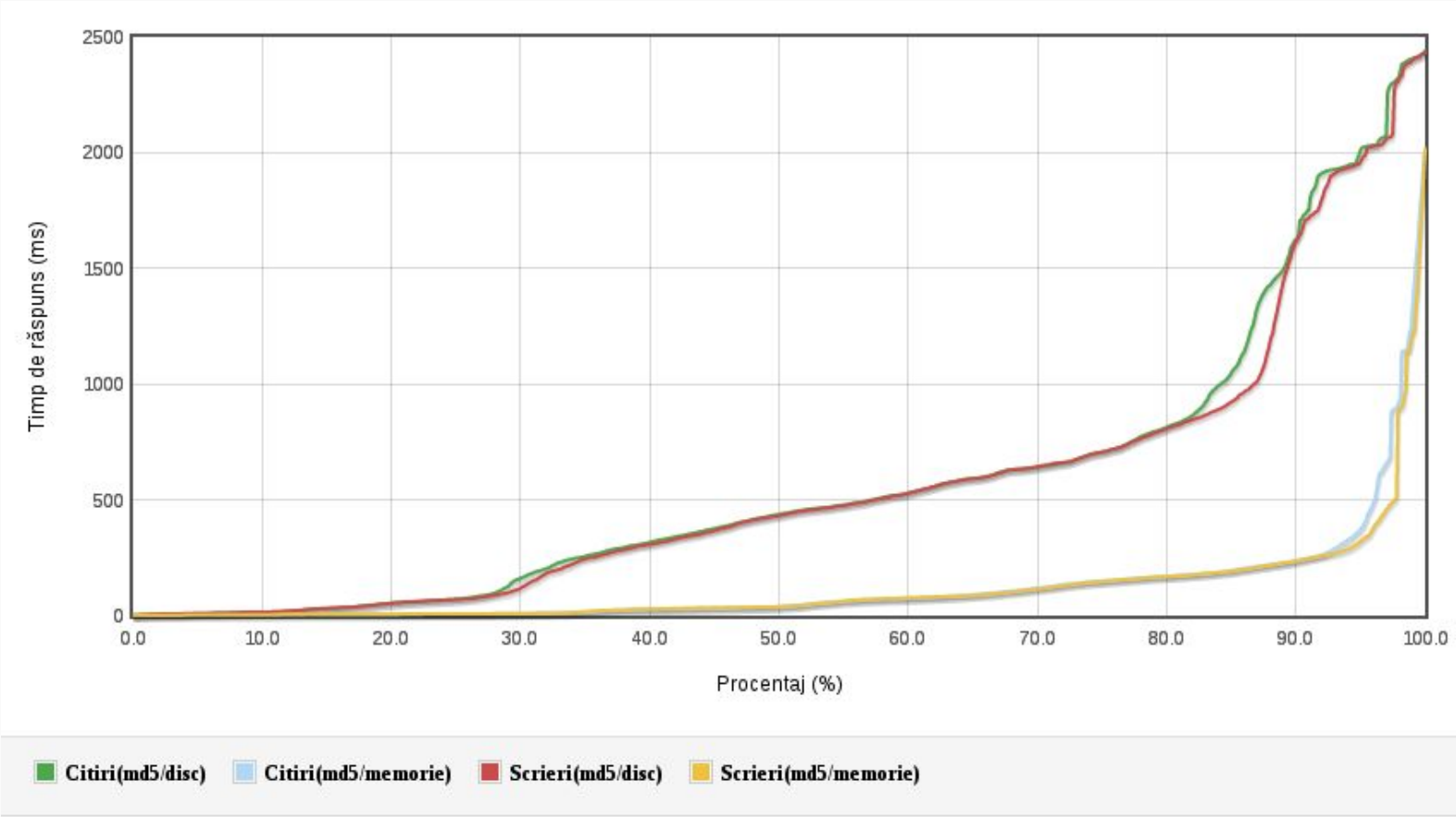
Infrastructură utilizată:

- 4 mașini virtuale pentru noduri
- 1 mașină virtuală pentru rularea testului de benchmark

Rezultate(crc32)

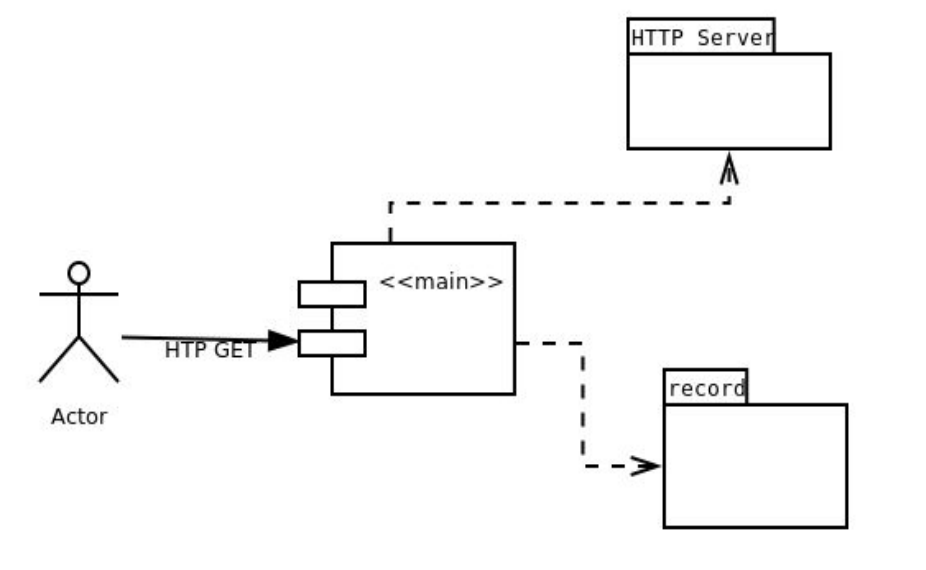


Rezultate (md5)



Interfața grafică de monitorizare

Descriere generală



- Arhitectură client server
- 2 module componente
  - HTTP server
  - stocare metrici

Funcționalitate oferită

Permite vizualizarea:

- distribuției cheilor
- numărului de citiri pe fiecare nod
- numărului de scrieri pe fiecare nod
- numărului de chei existente
- memoriei consumate

Concluzii și îmbunătățiri viitoare

Concluzii

- gossip oferă premisele construirii unor sisteme de stocare a datelor de capacități foarte mari
- folosirea limbajului GO oferă performanțe foarte bune (RAM și CPU)
- scrierea pe disc ramâne principalul factor de limitare al performanțelor

Îmbunătățiri viitoare

- folosirea unui protocol gossip adaptabil
- adăugarea de facilități precum autentificare și autorizare
- algoritmi mai performanți de distribuire a datelor

Contribuții personale

- explorarea folosirii sistemelor distribuite de tip gossip
- explorarea folosirii algoritmilor de tip consistent hashing și implementare
- implementarea unui asemenea prototip

Mulțumesc !