Algorytm Raft

Problem konsensusu

Czym jest Raft?

- Algorytm wykorzystywany w problemie konsensusu
- Zaprojektowany w sposób łatwy do zrozumienia
- Równoważny do algorytmu Paxos pod względem odporności na błędy oraz wydajności
- Dekompozycja do osobnych podproblemów
- Uwzględnia problemy w rzeczywistych systemach

Algorytm Raft - idea

Problem konsensusu to jeden z fundamentalnych filarów rozproszonych systemów odpornych na błędy.

Pomysłem na rozwiązanie jest kontekst w postaci zreplikowanych maszyn stanów, gdzie każdy węzeł posiada maszynę stanów i zbiór logów.

Maszyna stanów to komponent, który powinien być odporny na błędy.

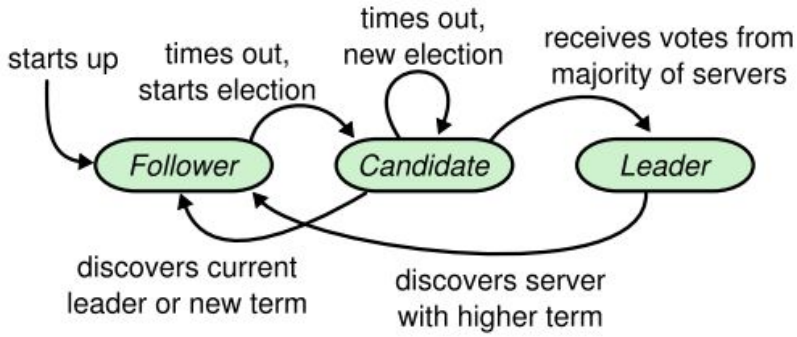
Dla klientów rozproszona maszyna stanów powinna zachowywać się jak pojedynczy, niezawodny komponent, nawet gdy kilka serwerów w klastrze zostanie wyłączonych.

Algorytm Raft - założenia

1. **“Electron Safety”** - dokładnie jeden lider może zostać wybrany w danym przedziale czasowym
2. **“Leader Append-Only”** - lider grupy nigdy nie nadpisuje i nie usuwa wpisów w logu, może jedynie je dodawać
3. **“Log Matching”** - jeśli dwa osobne logi posiadają dany wpis z takim samym indeksem i numerem przedziału czasowego, wtedy wszystkie logi w ich wpisach są poprawne aż do tego indeksu
4. **“Leader Completeness” -** jeśli dany wpis jest zatwierdzony w danym przedziale czasowym, to będzie on widoczny w logu wszystkich liderów dla następnych przedziałów czasowych
5. **“State Machine Safety”** - jeśli serwer dodał kolejny wpis pod danym indeksem do maszyny stanów, wtedy żaden inny serwer nie ma prawa dodać innego wpisu pod ten sam indeks

Stany węzłów

- Follower
- Candidate
- Leader



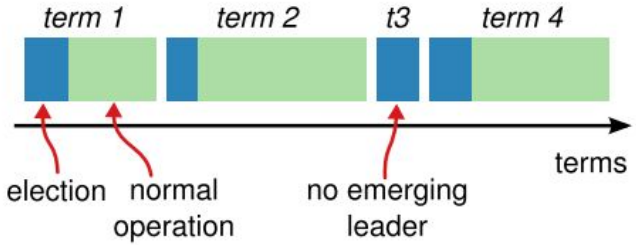
“Followers” odpowiadają wyłącznie na żądania innych serwerów. Jeśli wystąpi brak komunikacji, przechodzi w stan “Candidate”.

“Candidate” - inicjuje elekcję lidera grupy. Otrzymując głosy większości serwerów zostaje nowym liderem.

“Leader” - działają dopóki nie padną ;)

Raft “terms”

- Identyfikator obecnego przedziału czasowego i jego aktualizacja
- Maksymalnie jeden lider

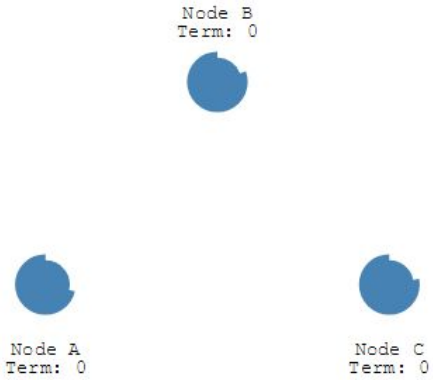


Czas jest podzielony na przedziały czasowe, z których każdy rozpoczyna się elekcją. Po udanej elekcji **leader** zarządza całym klastrem serwerów, aż “term” się nie skończy.

W przypadku nieprawidłowej elekcji, przedział czasowy może się zakończyć bez wybrania lidera grupy.

Elekcja lidera

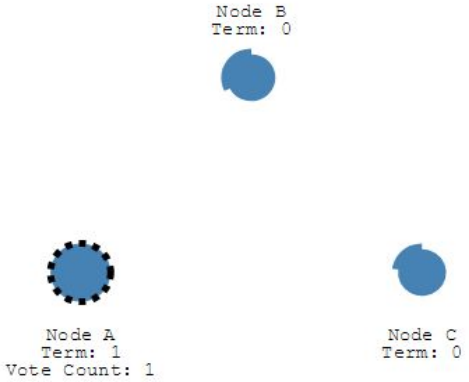
- Proces wyboru reprezentanta klastra
- Lider odpowiada za rozsyłanie poleceń od klienta i utrzymanie spójnego stanu



1. Każdy nowy węzeł oczekuje 150-300 ms zanim przejdzie w stan kandydata
2. Randomizowane czasy oczekiwania są kluczowe w procesie wyboru lidera w Raft

Elekcja lidera

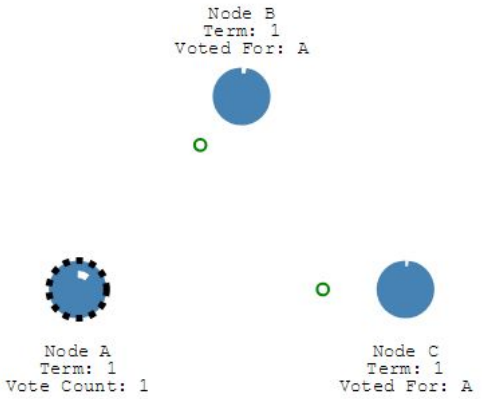
- Proces wyboru reprezentanta klastra
- Lider odpowiada za rozsyłanie poleceń od klienta i utrzymanie spójnego stanu



3. Węzeł przechodzi w stan Candidate...
4. ...głosuje na samego siebie…
5. ...po czym wysyła zdarzenia Request Vote do pozostałych węzłów

Elekcja lidera

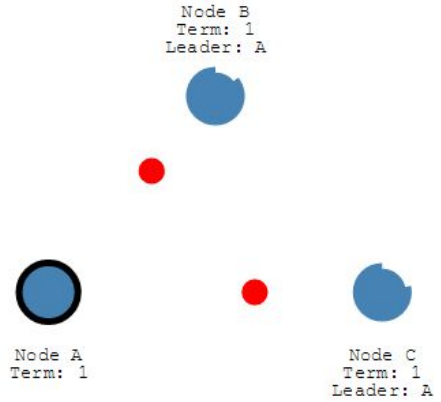
- Proces wyboru reprezentanta klastra
- Lider odpowiada za rozsyłanie poleceń od klienta i utrzymanie spójnego stanu



6. Jeśli węzeł w danej turze (term) jeszcze nie głosował, odpowiada na Request Vote
7. Jeśli dany węzeł (tu A) otrzyma większość głosów (kworum) zostaje liderem klastra
8. Reszta węzłów przechodzi w stan Follower

Elekcja lidera

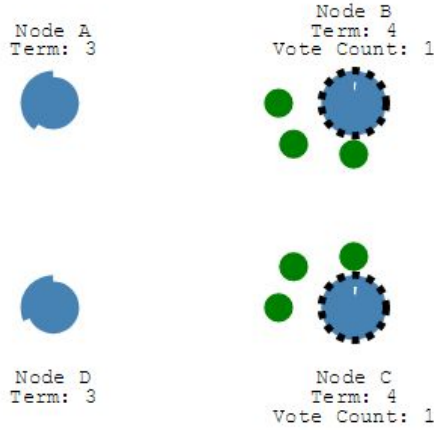
- Proces wyboru reprezentanta klastra
- Lider odpowiada za rozsyłanie poleceń od klienta i utrzymanie spójnego stanu



9. Lider co jakiś czas przypomina o swoim istnieniu rozsyłając wiadomości Heartbeat
10. Jeśli Follower przestanie odbierać Heartbeaty przez określony czas (150-300 ms), przechodzi w stan Candidate i rozpoczyna nowe wybory

Elekcja lidera - split vote

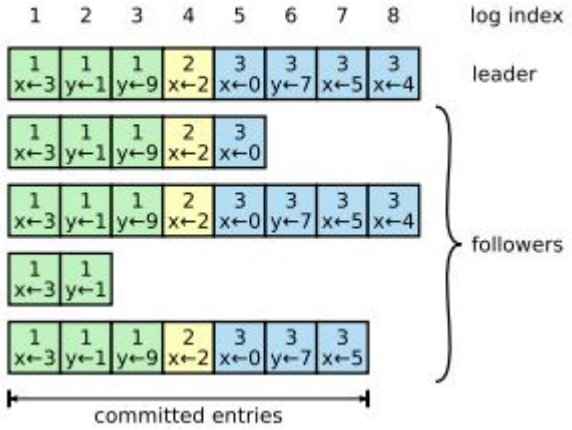
- Sytuacja, w której jednocześnie pojawia się wielu kandydatów i każdy dostaje część głosów, ale nie większość



1. Żaden z kandydatów nie otrzyma większości głosów
2. Stan Candidate również posiada timeout, po upływie którego staje się nowym kandydatem w kolejnej turze wyborów
3. Randomizacja timeoutu zapewnia, że proces kiedyś się skończy i lider zostanie wybrany

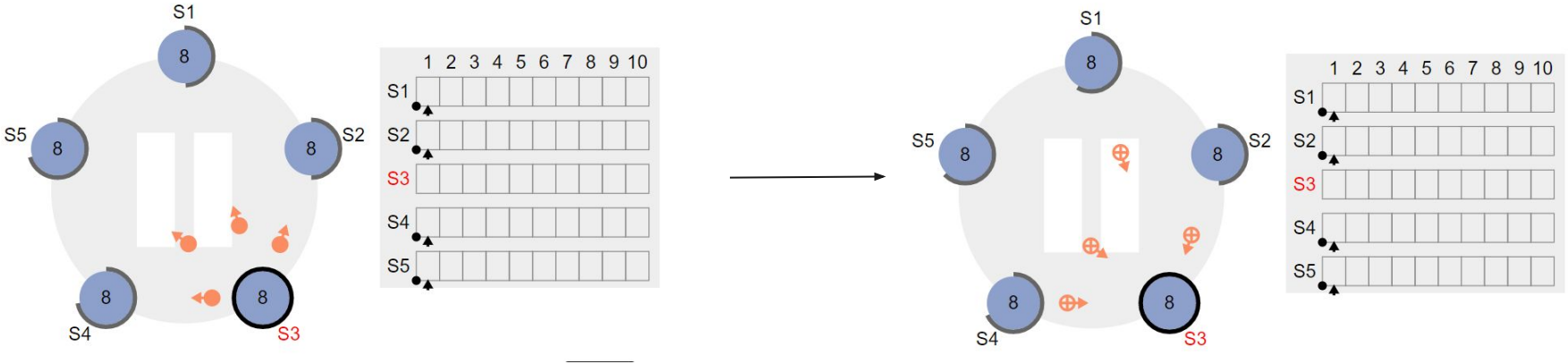
Replikacja logu

- Proces uzgadniania zgodnego stanu klastra
- Za utrzymanie spójności odpowiada Lider

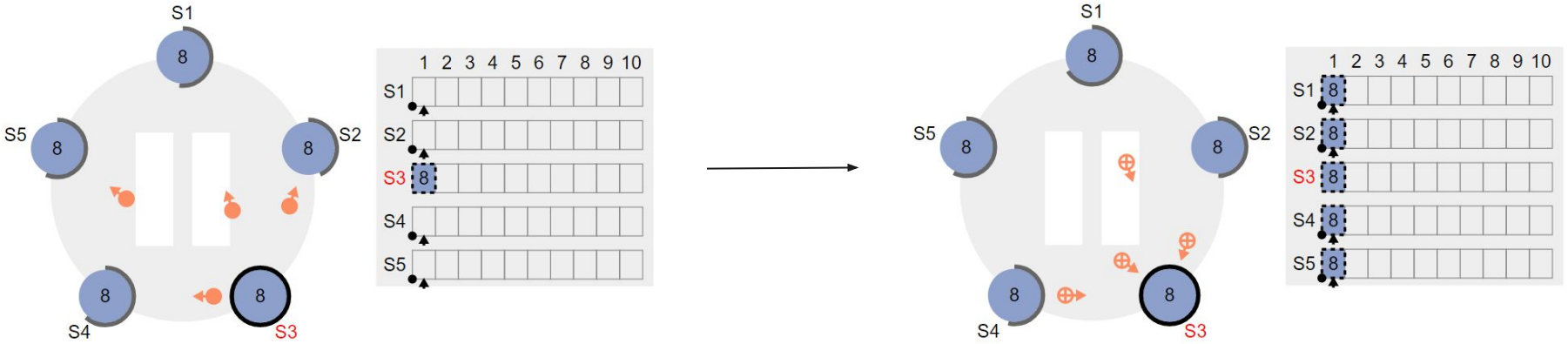


- Każdy węzeł ma kolejkę zdarzeń - log
- Zdarzenia składają się z numeru termu, w którym nadeszły oraz instrukcji do maszyny stanów (np. dodaj 3)
- Stan uznaje się za commited, jeśli Lider dostał potwierdzenie od większości węzłów

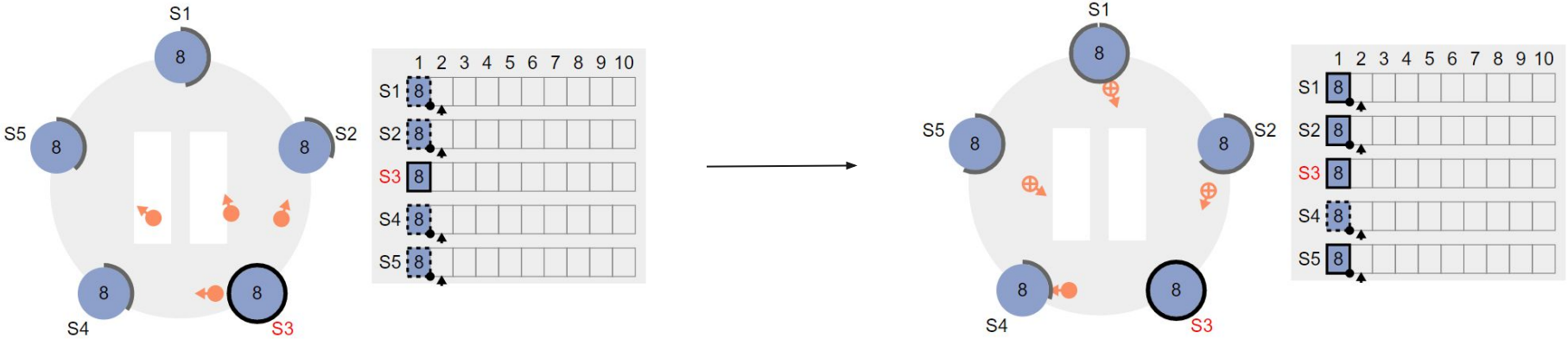
Wizualizacja - replikacja logu



Wizualizacja - replikacja logu

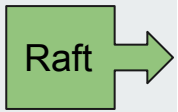


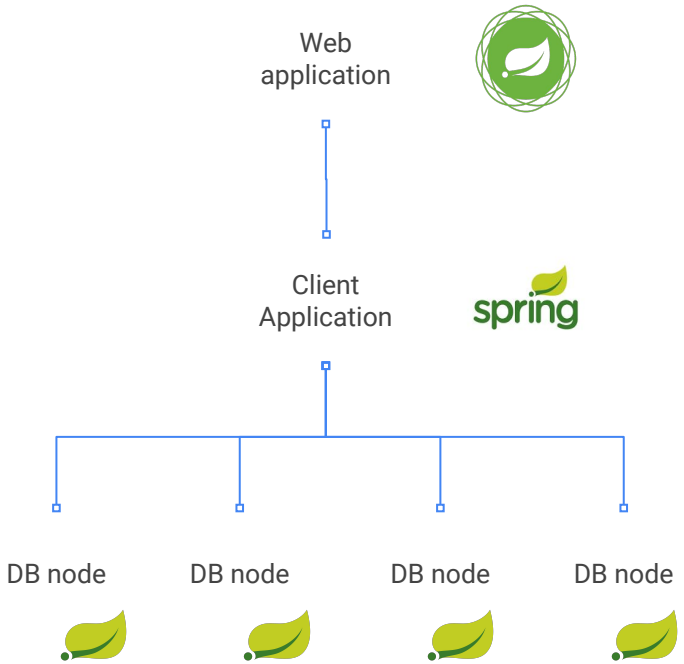
Wizualizacja - log replication



Implementacja replikowanej bazy danych

Architektura





Stos technologiczny

- **Spring Boot** - Java-based framework
- **Thymeleaf** - template engine
- **RabbitMQ** - message broker
- **Travis CI** - continuous integration service
- **Heroku** - platform as a service
- **Docker -** container technology
- **Jacoco, PMD** - static code analysis

Koniec

Autorzy:

- Łukasz Gajewski
- Paweł Kocot
- Grzegorz Siatka