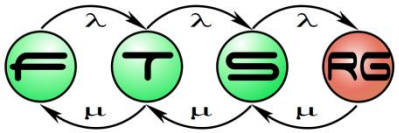
Elosztott adatbázis-kezelő formális elemzése

Szárnyas Gábor
szarnyas@mit.bme.hu

2014. december 10.

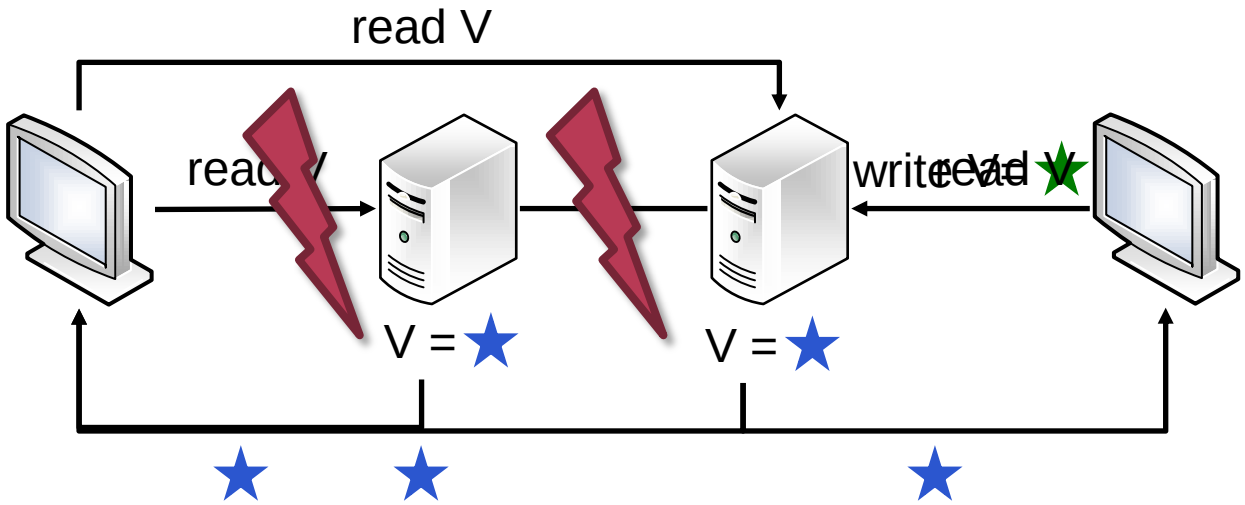
Budapesti Műszaki és Gazdaságtudományi Egyetem
Hibatűrő Rendszerek Kutatócsoport



ELOSZTOTT ADATKEZELÉS

Replikáció

- Többpéldányos tárolás



Elméleti korlát: a CAP tétel

- Sejtés: Eric Brewer, 2000
- Tétel: Nancy Lynch, Seth Gilbert, 2002

Fischer–Lynch–Paterson lehetetlenségi tétel

- Tulajdonságok:
  - **C**onsistency
  - **A**vailability
  - **P**artition tolerance

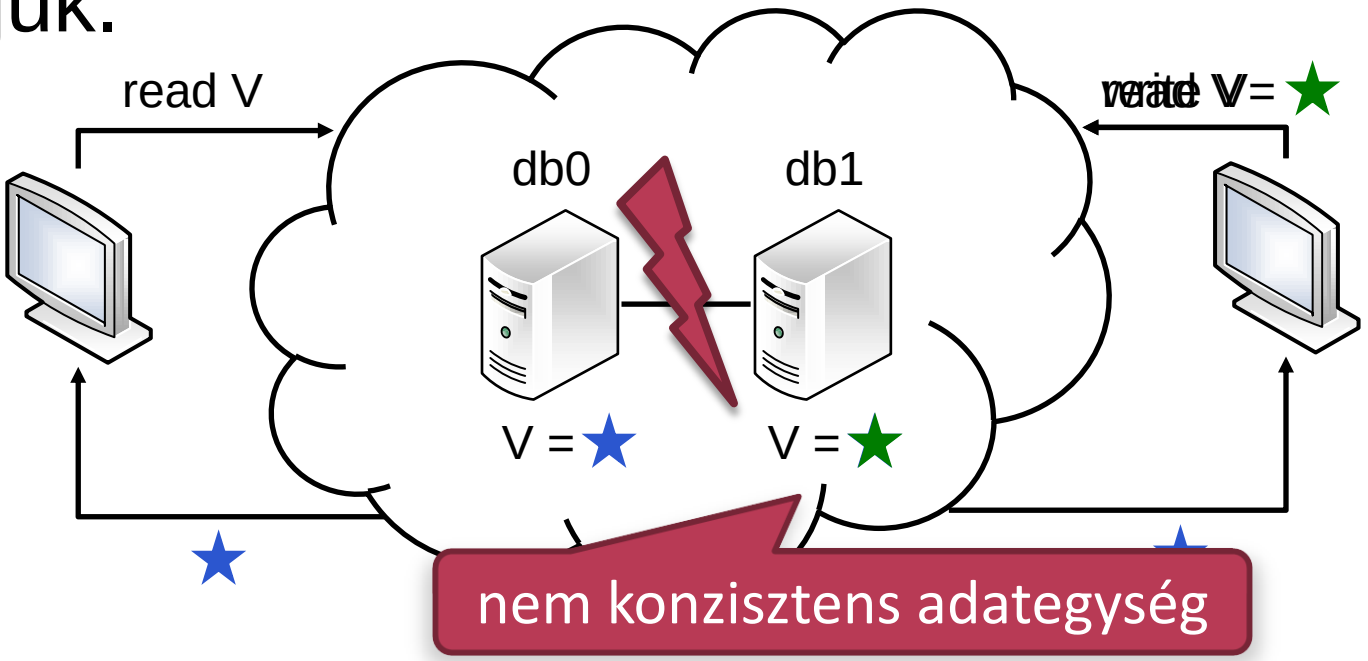
- **Sejtés.** Elosztott rendszerben egy időben nem garantálható mindhárom tulajdonság.

CAP tétel precízen

- **Tétel.** Egy elosztott rendszerben nem biztosítható, hogy a rendszer *mindig* (üzenetek elvesztése esetén is) garantálja az alábbi tulajdonságokat:
  - atomi konzisztencia (**c**onsistency),
  - rendelkezésre állás (**a**vailability).
- **Bizonyítás.** Seth Gilbert, Nancy Lynch, *Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services,* http://dl.acm.org/citation.cfm?id=564601

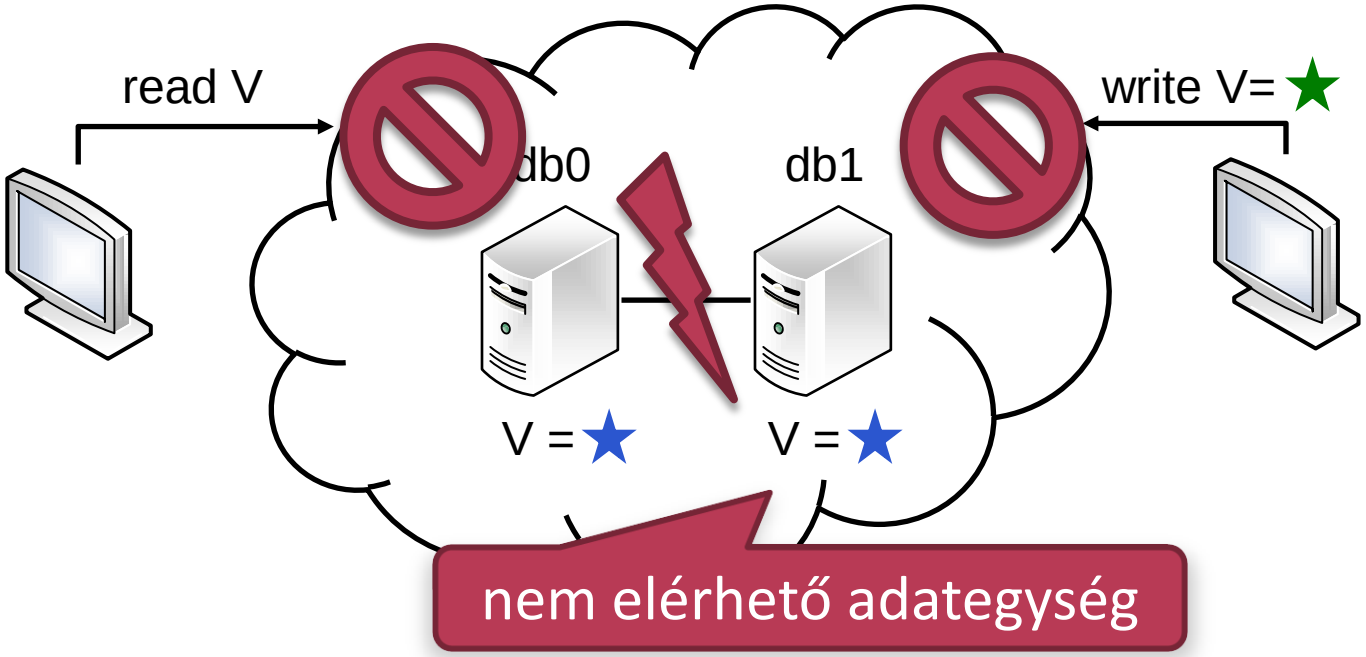
Consistency – konzisztencia

- Egy adategység értékét bármely csomóponttól lekérdezve ugyanazt az értéket kapjuk.



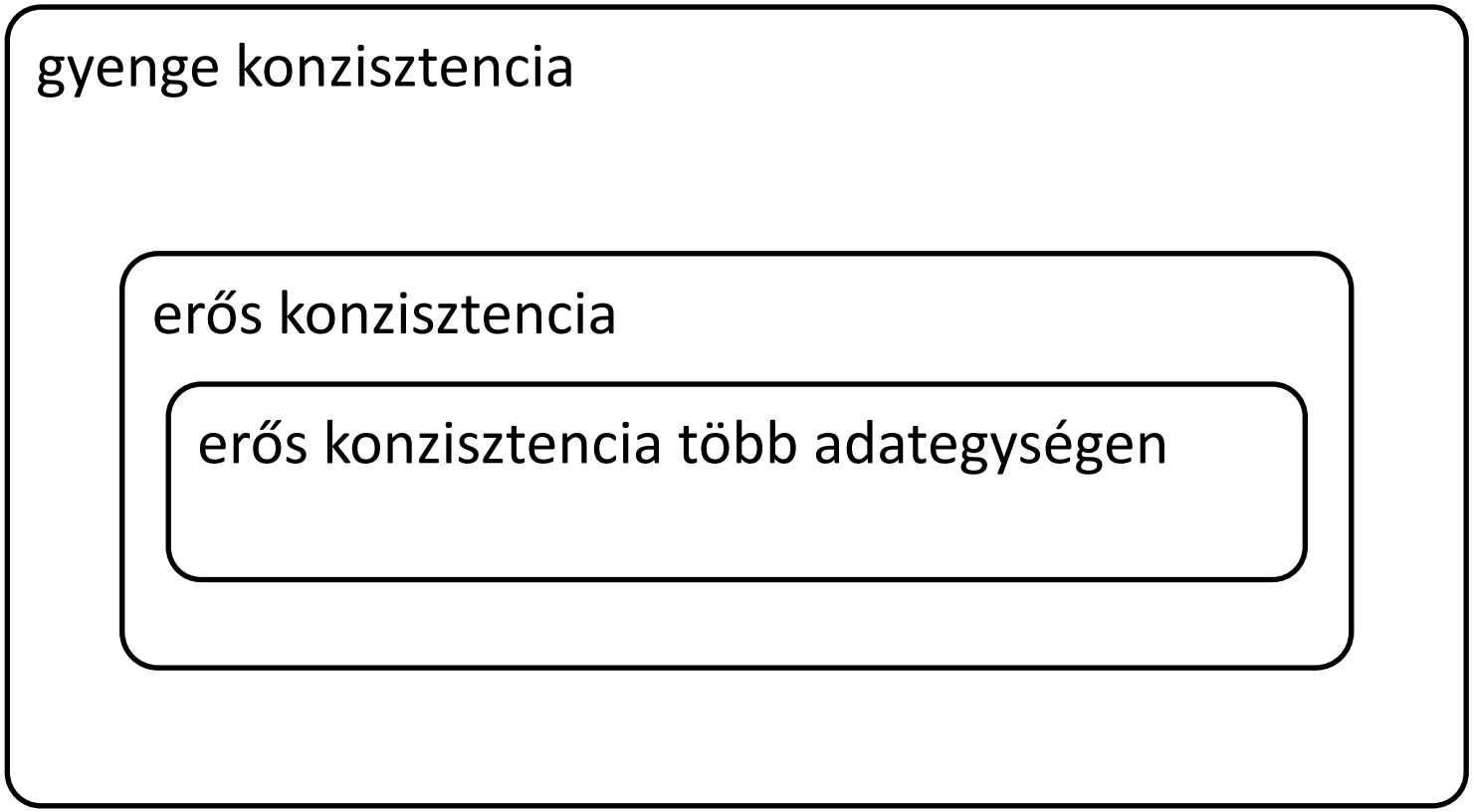
Availability – rendelkezésre állás

- A rendszer minden működő csomóponthoz érkező kérésre válaszol



Konzisztenciamodellek

- A CAP tétel következménye



Gyenge konzisztencia – YouTube

Learn MapReduce with Playing Cards

by Jesse Anderson

18,858 views

nincs szükség pontos értékre

Learn MapReduce with Playing Cards

Jesse Anderson

Subscribe 177

19,023

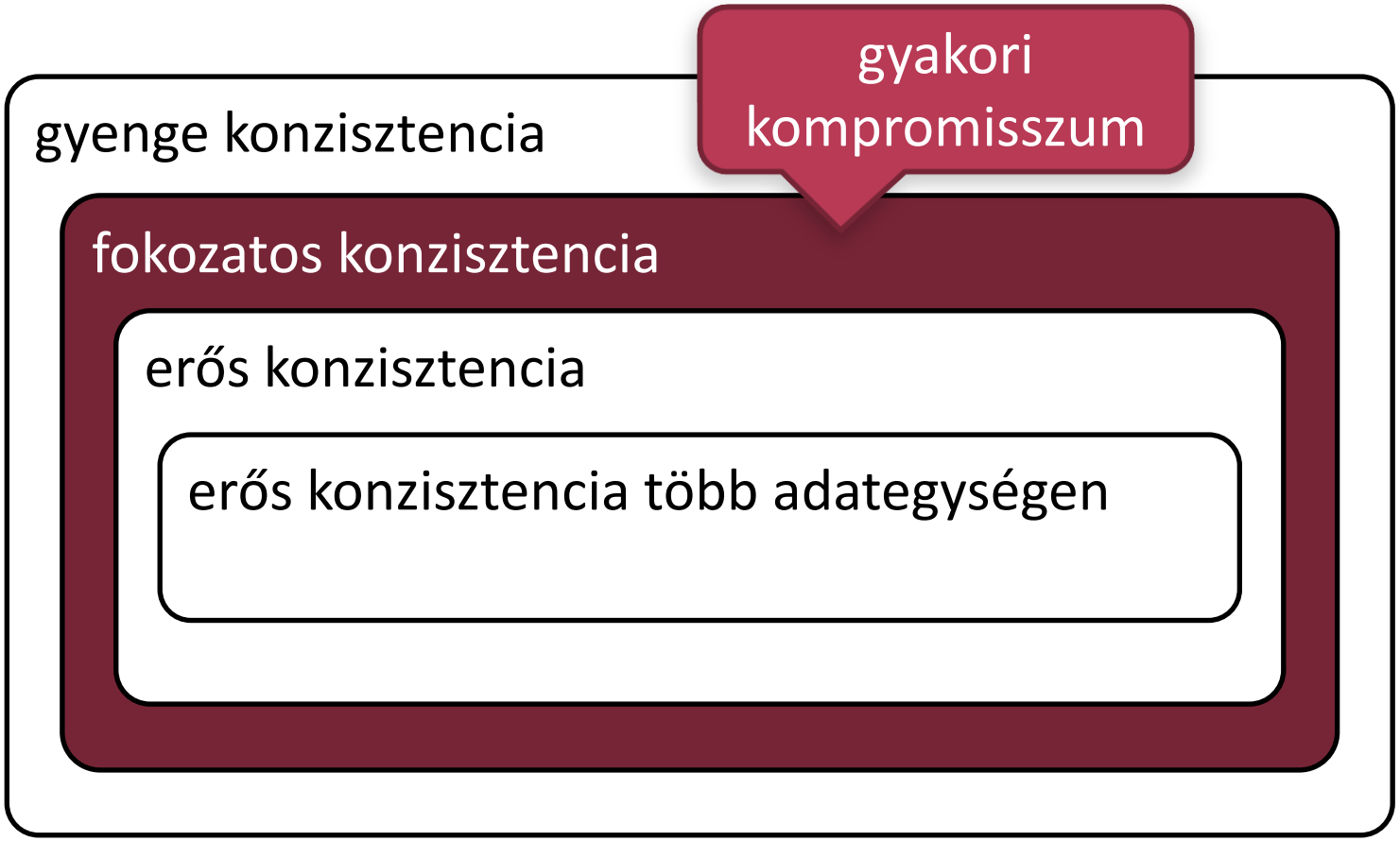
Add to Share More 161 7

Published on Aug 14, 2013

The special extended preview of my new MapReduce screencast available for purchase at http://pragprog.com/screencasts/v-jam....

Konzisztenciamodellek

- A CAP tétel következménye



Fokozatos konzisztencia – Tumblr

Tranzakciók – ACID garanciák

- Atomicity
- Consistency
- Isolation
- Durability

http://en.wikipedia.org/wiki/Acid_test_(gold)

ACID–BASE spektrum

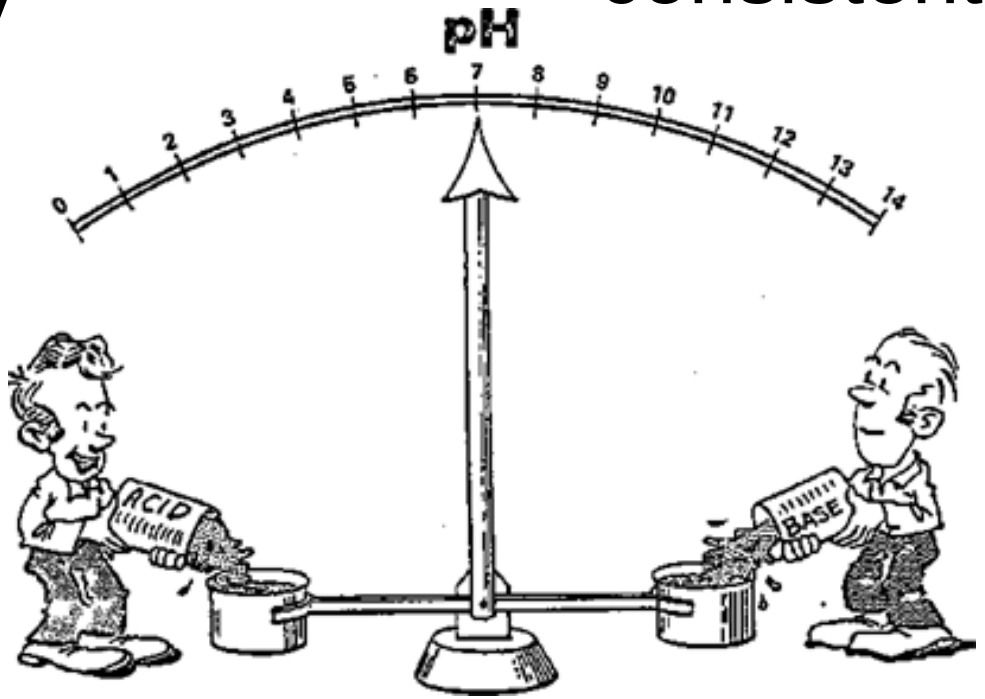
- Atomicity
- Consistency
- Isolation
- Durability

- Basically Available
- Soft state
- Eventually consistent

?

?

?



CASSANDRA

Cassandra

- 2007 – Dynamo (Amazon)
- 2008 – Cassandra (Facebook)
- 2009 – Apache projekt

NETFLIX

Spotify

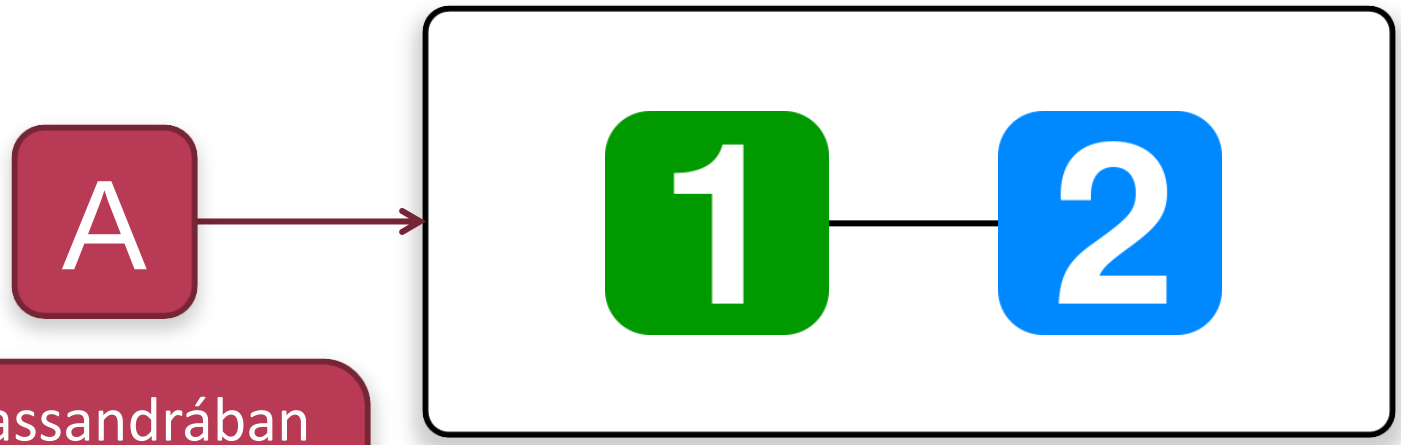
ebay

DB-Engines Ranking – 2014. dec.

Rank	Last Month	DBMS	Database Model	Score	Changes
1.	1.	Oracle	Relational DBMS	1459.79	+7.67
2.	2.	MySQL	Relational DBMS	1268.58	-10.50
3.	3.	Microsoft SQL Server	Relational DBMS	1200.05	-20.15
4.	4.	PostgreSQL	Relational DBMS	254.01	-3.35
5.	5.	MongoDB	Document store	246.52	+1.78
6.	6.	DB2	Relational DBMS	210.25	+4.02
7.	7.	Microsoft Access	Relational DBMS	139.89	+1.06
8.	8.	SQLite	Relational DBMS	94.70	-0.58
9.	9.	Cassandra	Wide column store	94.06	+2.07
10.	↑ 11.	Redis	Key-value store	87.88	+5.53

http://db-engines.com/en/ranking

Konzisztenciaszintek



A Cassandrában **műveletenként** specifikálható

$f = 2$ esetén

- ONE: 1 $\rightarrow 1$
- QUORUM: $\left\lfloor \frac{f}{2} + 1 \right\rfloor$ $\rightarrow 2$
- ALL: f $\rightarrow 2$

Konzisztencia ellenőrzése

1. Szisztematikus terhelés
   - Adatbázis és naplófájlok elemzése
2. Implementáció helyességének ellenőrzése
   - A gyakorlatban kivitelezhetetlen

Cassandra formális elemzése

- Si Liu, et. al,
  Formal Modeling and Analysis of Cassandra in Maude,
- Tech report:
  https://sites.google.com/site/siliunobi/icfem-cassandra

- Eszköz: **Maude**
- LTL modellellenőrző
  - Adott állapotból kiindulva teljesül-e egy temporális logikai kifejezés

Kapcsolódó munkák

- Adatbázis-kezelők formális verifikációja
  - NoSQL rendszereknél nincs
  - Megastore (Google)
- Felhő alapú rendszerek verifikációja
  - ZooKeeper – rendelkezésre állás
  - DoS resilience mechanizmusok
  - KLAIM nyelv – architektúra elemzés

Konzisztenciamodellek formálisan

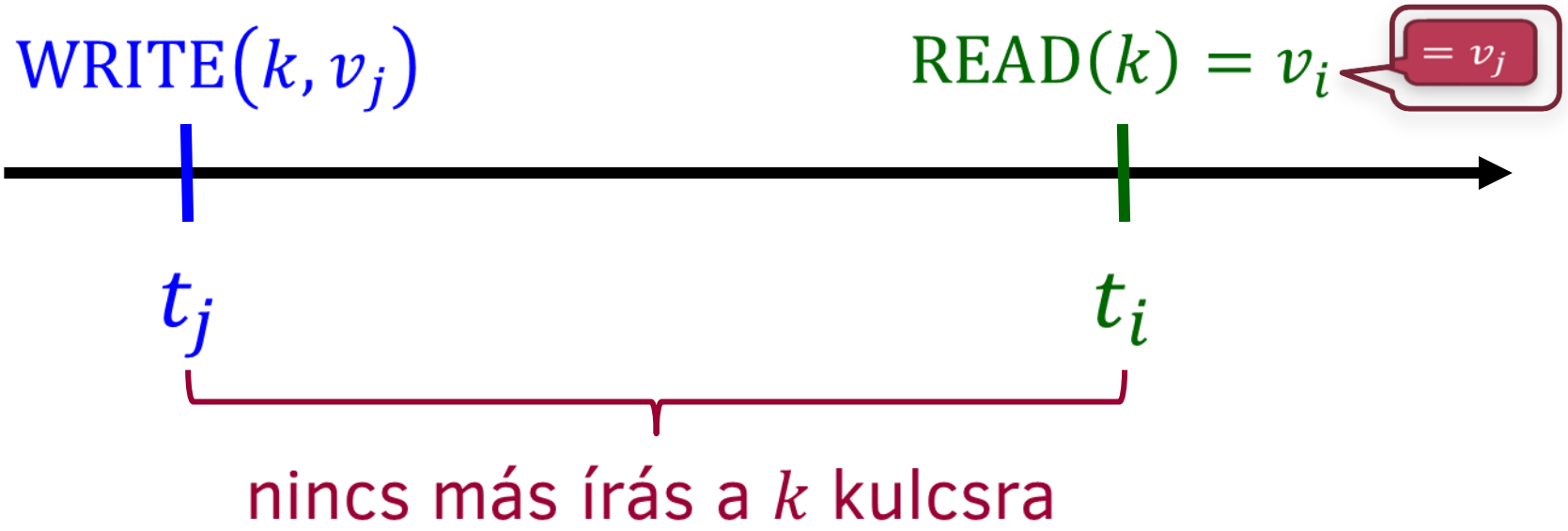
- Írás és olvasás műveletek trace-e:
  - $Tr = o_1, o_2, \ldots, o_n$ műveletek sorozata,
  - ahol $o_i = (k, v, t)$
- Ellenőrzés egy k adategységen

Erős konzisztencia

Ha egy $o_i = (k, v_i, t_i)$ olvasásra

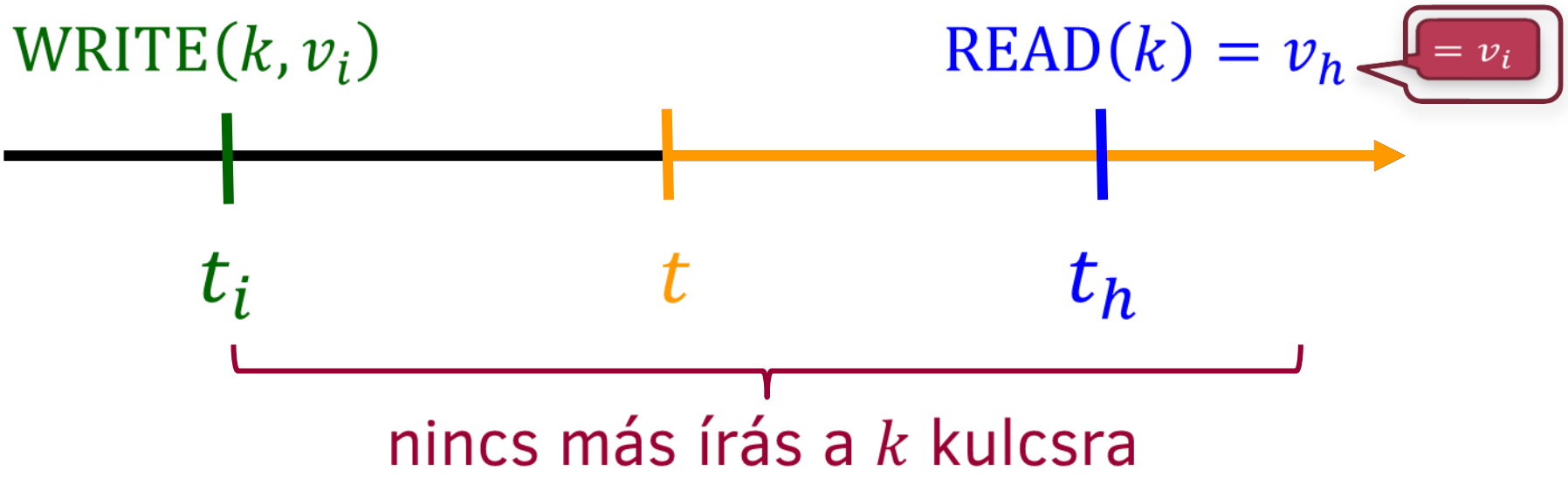
- $\exists o_j = (k, v_j, t_j)$ írás, ahol $t_j < t_i$ és
- $\nexists o_h = (k, v_h, t_h)$ írás, ahol $t_j < t_h < t_i$,

akkor $v_i = v_j$.

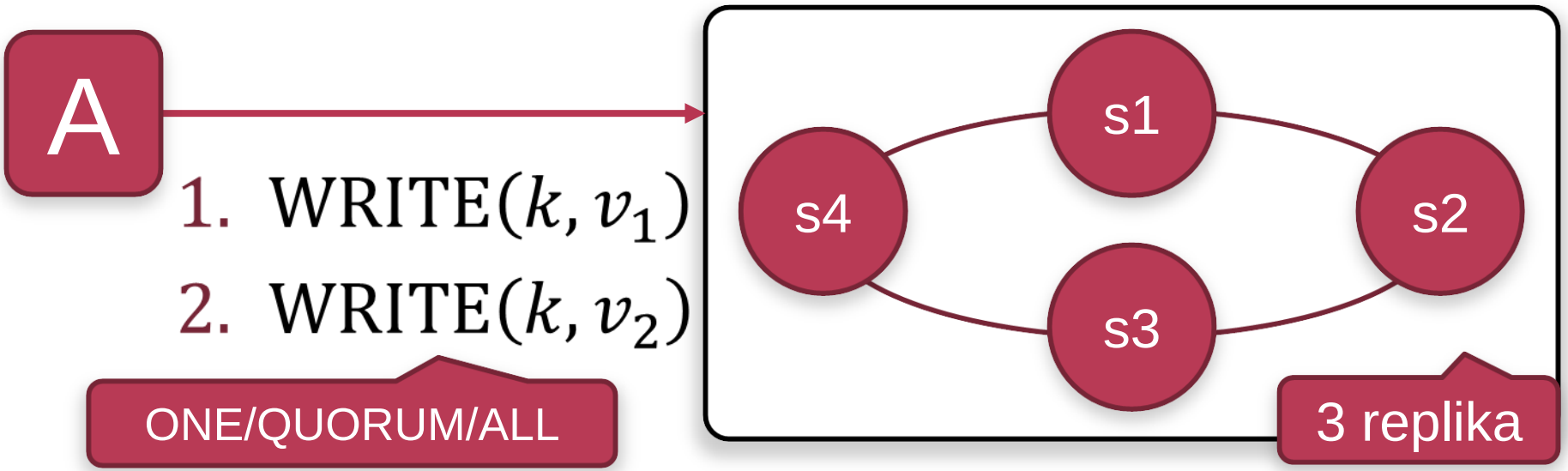


Fokozatos konzisztencia

Legyen $o_i = (k, v_i, t_i)$ egy írás
és $\nexists o_j = (k, v_j, t_j)$ írás, ahol $t_i < t_j$.
$\exists t > t_i$,
hogy $\forall o_h = (k, v_h, t_h), t_h > t$ olvasásra
$v_h = v_i$.



Fokozatos konzisztencia ellenőrzése - 1 kliens




```
op eventual : Address Address Address Key Value -> Prop .
eq < R1 : Server | table: (K |-> (V,T1), ...), ... >
   < R2 : Server | table: (K |-> (V,T2), ...), ... >
   < R3 : Server | table: (K |-> (V,T3), ...), ... >
REST |= eventual(R1,R2,R3,K,V) = true .
```

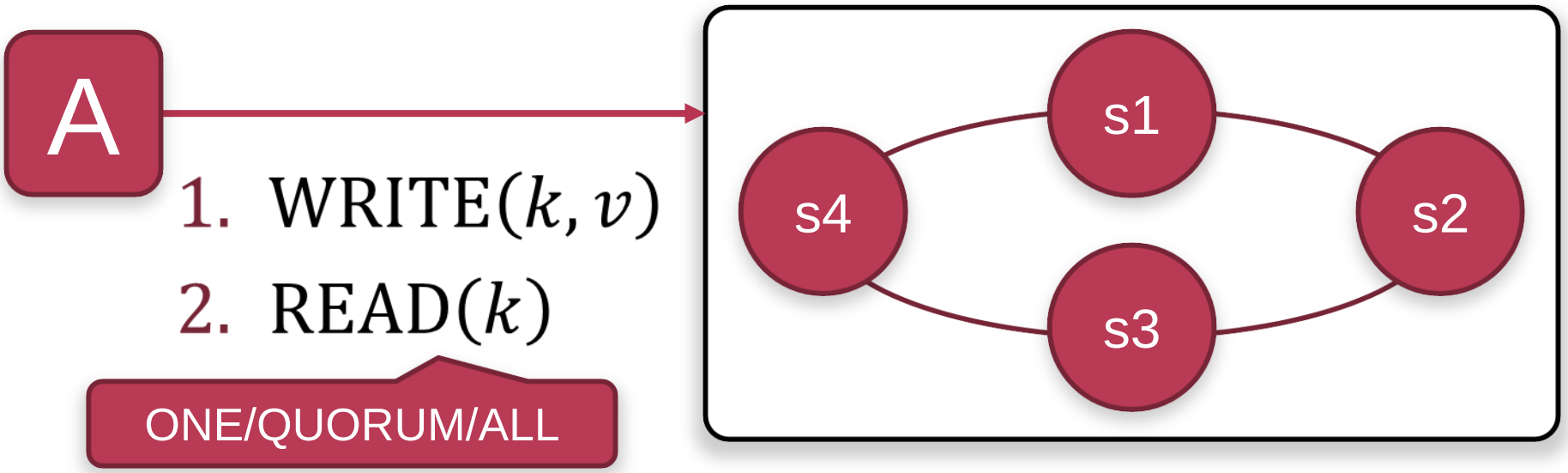
```
red modelCheck(initConfig,
<>[] eventual(r1,r2,r3,key,value)) .
```

Fokozatos konzisztencia

- Fokozatos konzisztencia: mindig garantált

$Write_1$ \ $Write_2$	ONE	QUORUM	ALL
ONE	✓	✓	✓
QUORUM	✓	✓	✓
ALL	✓	✓	✓

Erős konzisztencia ellenőrzése – 1 kliens




```
op strong : Address Value -> Prop .
  eq <A:Client | store:(ID,K,V), ...>
  REST |= strong(A,K,V) = true .
```

```
red modelCheck(initConfig,
<> strong(client,key,value)) .
```

Erős konzisztencia

- $\mathrm{WRITE}(k, v)$ ONE ONE QUORUM
- $\mathrm{READ}(k)$ ONE QUORUM ONE

Write_1 \ Read_2	ONE	QUORUM	ALL
ONE	×	×	✓
QUORUM	×	✓	✓
ALL	✓	✓	✓

Több klienssel

- Fokozatos konzisztencia garantált
- Erős konzisztencia sérülhet

Strong

Latency \ Consistency Lv.	ONE	QUORUM	ALL
L1 (L1<D1)	×	×	×
L2 (D1<L2<D2)	×	×	×
L3 (D2<L3)	✓	✓	✓

Eventual

Latency \ Consistency Lv.	ONE	QUORUM	ALL
L1 (L1<D1)	✓	✓	✓
L2 (D1<L2<D2)	✓	✓	✓
L3 (D2<L3)	✓	✓	✓

Összefoglalás

- Konzisztenciamodellek formális verifikációja megvalósítható
- A Cassandra modellje garantálja az ígért konzisztenciaszinteket
- Időzítési kényszerek további vizsgálata lehetséges (Real-Time Maude)