Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services

Presented by
Vikram Rao & Steven Guan
UC DAVIS

Prof. Brewer

Figure: Prof. Eric Brewer, University of California, Berkeley

Prof. Brewer's Talk

PODC 2000 Invited Talk

July 19, 2000

Towards Robust Distributed Systems

Eric A. Brewer

University of California, Berkeley

Motivation

Prof. Brewer's talk:

Current distributed systems, even the ones that work, tend to be very fragile: they are hard to keep up, hard to manage, hard to grow, hard to evolve, and hard to program

In this talk, I look at several issues in an attempt to clean up the way we think about these systems

...

Motivation

Prof. Brewer's talk:

These are not (yet) provable principles, but merely ways to think about the issues that simplify design in practice

They draw on experience at Berkeley and with giant-scale systems built at Inktomi, including the system that handles 50% of all web searches

...

Motivation

Figure: New Game APP

Motivation

Figure: Multi Player User Game APP

Motivation

Figure: Your MMORPG Game APP is awesome and more Users play!

Motivation

Figure: You scale up then Server crashes frequently, users complaint server issues and you don't know how to solve this issue perfectly!

?? Provide users 24/7 game run despite high load ??

?? Provide users latest game updates ??

?? Equally accessible to users across globe ??

Contents

Atomic Data Objects

There must exist a total order on all operations such that each operation looks as if it were completed at a single instant. for ex: return the right response to each request.

Available Data Objects

Every request received by a non-failing node in the system must result in a response. for ex: any service must eventually terminate

Partition Tolerance

When the network is partitioned all messages sent from nodes in one partition to nodes in another partition are lost

CAP vs ACID

Figure: CAP is not made of ACID!

Contents

Networks – Asynchronous

It is impossible in the asynchronous network model to implement a read/write data object that guarantees the following properties:

- Availability
- Atomic consistency

in all fair executions (including those in which messages are lost).

Networks – Asynchronous

- We prove this by contradiction
- Assume that the network consists of at least two disjoint nodes N1, N2
- Assume network meets the three criteria: CAP
- All messages between N1 and N2 are lost

Networks – Asynchronous

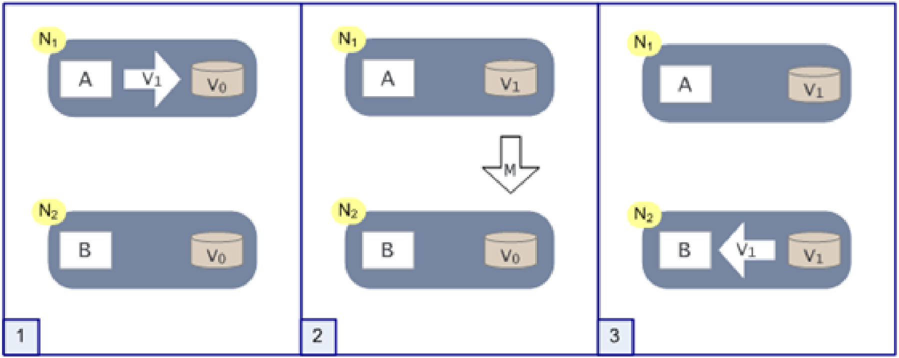


Figure: Best case scenario

Networks – Asynchronous

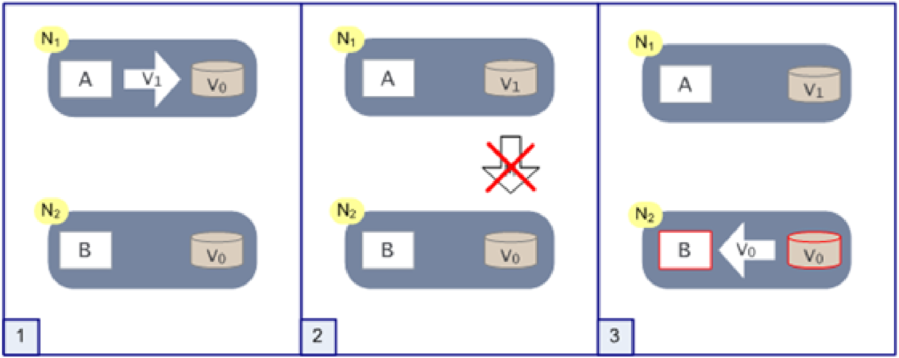
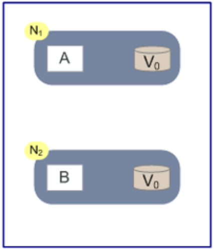


Figure: Worst case scenario



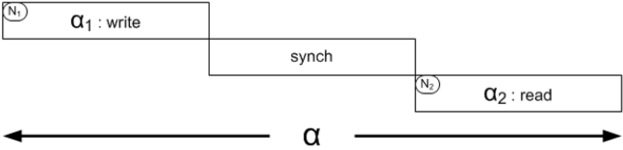


Figure: Case where messages aren't lost

Networks – Partially Synchronous

It is impossible in the partially synchronous network to implement a read/write data object that guarantees the following properties:

- Availability
- Atomic consistency

in all fair executions (including those in which messages are lost).

Networks – Partially Synchronous

- We prove this by contradiction
- Assume that the network consists of at least two nodes G1, G2
- Assume network meets the three criteria: CAP
- All messages between G1 and G2 are lost

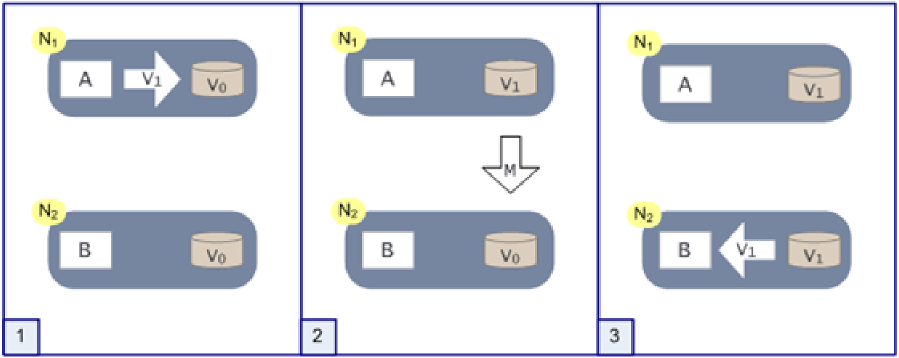


Figure: Best case scenario

Networks – Partially Synchronous

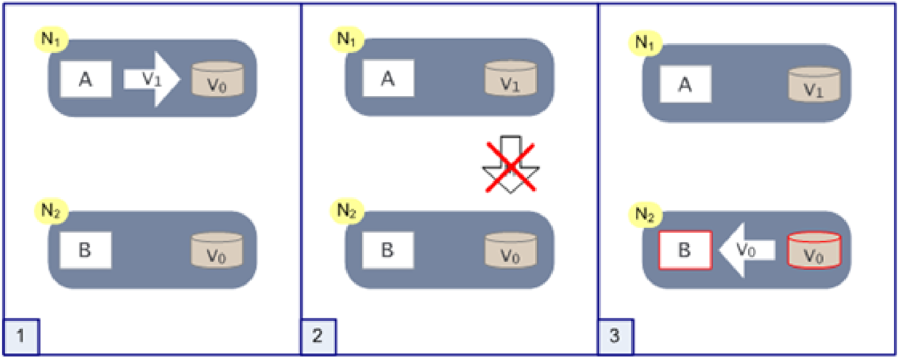


Figure: Worst case scenario

Partially Synchronous Networks

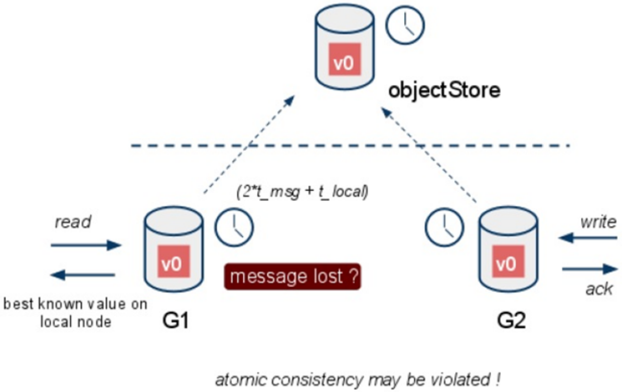


Figure: Centralized Solution

Solution – Take 2

Figure: Pick any 2 and start designing!

Solution – Weaker Consistency Conditions

This guarantee allows for some stale data when messages are lost, but provides a time limit on how long it takes for consistency to return, once the partition heals.

Definition – t-consistent

A timed execution of a read-write object is t-Connected Consistent if two criteria hold. First in executions in which no messages are lost, the execution is atomic. Second, in executions in which messages are lost, there exists a partial order P on the operations.

Centralized algorithm

• What is centralized algorithm? • Is the modified centralized algorithm is t-Connected consistent?

Solution – Partition Decision

cancel the operation and thus decrease availability,
or
proceed with the operation and thus risk inconsistency

Contents

Conclusion

- Proved that CAP is impossible to reliably provide atomic consistent data when there are partitions in the network
- Possible solutions with any two properties of C, A and P

Conclusion

Figure: Multi Player User Game APP

Contents

References

[1] Hagit Attiya, Amotz Bar-Noy, Danny Dolev, Daphne Koller, David Peleg, and Rüdiger Reischuk. *Achievable cases in an asynchronous environment.* 28th Annual Symposium on Foundations of Computer Science, Los Angeles, CA, 1987

[2] Eric A. Brewer, *Towards robust distributed systems*, Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing, Portland, Oregon, 2000

[3] Maurice P. Herlihy , Jeannette M. Wing, *Linearizability: a correctness condition for concurrent objects*, ACM Transactions on Programming Languages and Systems, 1990

[4] Leslie Lamport. *On interprocess communication – parts I and II. Distributed Computing*, 1986.

[5] Nancy A. Lynch, *Distributed Algorithms*, Morgan Kaufmann Publishers Inc, San Francisco, CA, 1996

Thank you

thank you
danke
рахмат
謝謝
ngiyabonga
شكراً جزيلاً
Баярлалаа
teşekkür ederim
спасибо
faafetai lava
mersi
barka
welalin
tack
spas
kia ora
vinaka
спасибі
blagodaram
dank je
misaotra
matondo
paldies
grazzi
mahalo
tapadh leat
kiitos
dankie
dhanyavad
gracias
хвала
asante
manana
obrigada
tenki
murakoze
bedankt
nanni
nandri
bayarlalaa
gràcie
hvala
mauruuru
köszönöm
dankon
aciū
akun
djiere dieuf
tau
mochchakkeram
дякую
mamnun
enkosi
dziękuję
chnorakaloutioun
gratias ago
gràcies
sulpáy
taiku
go raibh maith agat
chokrane
sobodi
dėkui
obrigado
sukriya
kop khun krap
ありがとう
arigatô
takk
dakujem
trugarez
mèsi
didi madloba
sagolun
najis tuke
terima kasih
tanemirt
rahmet
grazie
ευχαριστώ
diolch
dhanyavadagalu
shukriya
mercè
мерси
kam sah hamnida
rahmat
তোমাকে ধন্যবাদ
감사합니다
xièxie
merci