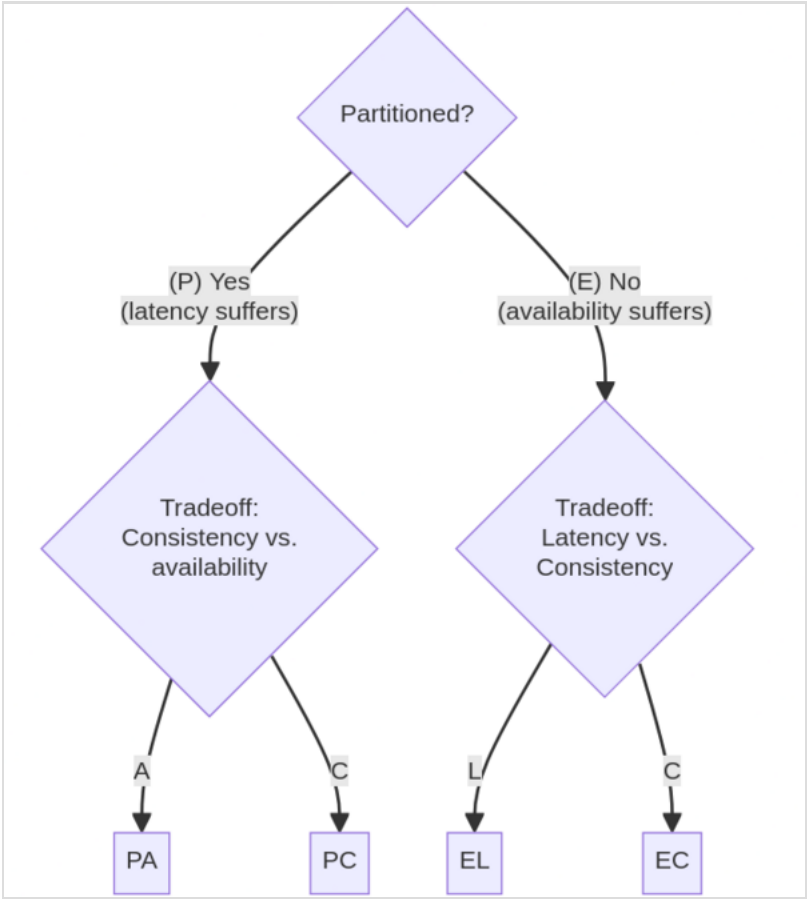
WIKIPEDIA

Teorema di PACELC

Nella teoria dei database, il **teorema PACELC** è un'estensione del teorema della CAP. Afferma che in caso di partizionamento di rete (P) in un sistema informatico distribuito, si deve scegliere tra disponibilità (A) e coerenza (C) (come per il teorema CAP), ma altrimenti (E), anche quando il sistema è in esecuzione normalmente in assenza di partizioni, si deve scegliere tra latenza (L) e perdita di coerenza (C).



Il compromesso tra disponibilità, coerenza e latenza, come descritto dal teorema PACELC.

La panoramica

Il teorema CAP può essere formulato come "PAC", il teorema di impossibilità che nessun archivio di dati distribuito possa essere sia coerente che disponibile in esecuzioni che contenga partizioni. Questo può essere dimostrato esaminando la latenza: se un sistema garantisce la coerenza, le latenze di funzionamento crescono con ritardi dei messaggi e quindi le operazioni non possono terminare alla fine se la rete è suddivisa, cioè il sistema non può garantire la disponibilità[.

In assenza di partizioni, sia la coerenza che la disponibilità possono essere soddisfatte[.[2] PACELC va quindi oltre ed esamina come il sistema replica i dati. In particolare, in assenza di partizioni, esiste un ulteriore compromesso (ELC) tra latenza e coerenza.[[3] Se il negozio è atomicamente coerente, allora la somma del ritardo di lettura e scrittura è almeno il ritardo del messaggio. In pratica, la maggior parte dei sistemi si basa su riconoscimenti espliciti piuttosto che su ritardi a tempo per garantire la consegna, richiedendo un viaggio di andata e ritorno di rete completo e quindi il ritardo del messaggio sia sulle letture che sulle scritture per garantire la coerenza[.[1] Nei sistemi a bassa latenza, al contrario, la coerenza è allentata al fine di ridurre la latenza[.[2]

Ci sono quattro configurazioni o compromessi nello spazio PACELC:

- PA/EL - dare priorità alla disponibilità e alla latenza rispetto alla coerenza
- PA/EC - quando c'è una partizione, scegliere la disponibilità; altrimenti, scegliere la coerenza
- PC / EL - quando c'è una partizione, scegliere la coerenza; altro, scegliere la latenza
- PC/EC - scegliere la coerenza in ogni momento

PC/EC e PA/EL forniscono modelli cognitivi naturali per uno sviluppatore di applicazioni. Un sistema PC/CE fornisce una garanzia ferma di coerenza atomica, come nell'ACID, mentre PA/EL fornisce alta disponibilità e bassa latenza con un modello di coerenza più complesso. Al contrario, i sistemi PA/CE e PC/EL rendono solo le garanzie condizionali di coerenza. Lo sviluppatore deve ancora scrivere codice per gestire i casi in cui la garanzia non è rispettata. I sistemi PA / CE sono rari al di fuori del settore della rete dati in memoria, dove i sistemi sono localizzati alle regioni geografiche e la latenza rispetto al compromesso di costanza non è significativa.[[4] Il PC / EL è ancora più difficile da capire. PC non indica che il sistema è completamente coerente; indica piuttosto che il sistema non riduce la coerenza oltre il livello di consistenza di base quando si verifica una partizione di rete, ma riduce la disponibilità.[3]

Alcuni esperti come Marc Brooker sostengono che il teorema CAP è particolarmente rilevante in ambienti connessi a intermittenza, come quelli relativi all'Internet of Things (IoT) e alle applicazioni mobili. In questi contesti, i dispositivi possono essere partizionati a causa di condizioni fisiche difficili, come interruzioni di corrente o quando si entra in spazi ristretti come gli ascensori. Per i sistemi distribuiti, come le applicazioni cloud, è più appropriato utilizzare il teorema PACELC, che è più completo e considera compromessi come la latenza e la coerenza anche in assenza di partizioni di rete.[5]

La storia

Il teorema PACELC è stato descritto per la prima volta da Daniel Abadi dall'Università di Yale nel 2010 in un post sul blog, [che in seguito ha chiarito in un documento nel 2012.[[3] Lo scopo di PACELC è quello di affrontare la sua tesi che "Ignorare la consistenza / latenza dei sistemi replicati è una grande supervisione [in CAP], in quanto è presente in ogni momento durante il funzionamento del sistema, mentre la CAP è rilevante solo nel caso della partizione. Il teorema PACELC è stato formalmente dimostrato nel 2018 in un articolo di SIGACT News[. [1]

Database PACELC valutazioni

[[3] Le valutazioni del database originale PACELC provengono da.[[6] Aggiornamenti successivi forniti dalla comunità di wikipedia.

- Le versioni predefinite della DB di Dynamo, (https://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf) Cassandra, Riak e Cosmos di Amazon (https://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf) sono sistemi PA / EL: se si verifica una partizione, rinunciano alla coerenza per la disponibilità e in condizioni di funzionamento normali rinunciano alla coerenza per una latenza inferiore.
- Sistemi completamente ACID come VoltDB / H-Store, Megastore, MySQL Cluster e

PostgreSQL sono PC / CE: si rifiutano di rinunciare alla coerenza e pagheranno i costi di disponibilità e latenza per raggiungerlo. Anche i sistemi Bigtable e correlati come HBase sono PC/CE.

- Amazon DynamoDB (lanciato nel gennaio 2012) è molto diverso dalla prima (Amazon internal) Dynamo (https://www.allthingsdistributed.com/files/amazon-dynamo-sosp2007.pdf) che è stata considerata per il documento PACELC. [[1] DynamoDB segue un forte modello di leader, in cui ogni scrittura è rigorosamente serializzata (e le scritture condizionali non comportano penalità) e supporta la coerenza di lettura dopo-scrittura. Questa garanzia non si applica alle "Tavole globali[[7]" in tutte le regioni. Gli SDK DynamoDB utilizzano alla fine letture coerenti per impostazione predefinita (miglioramento della disponibilità e del throughput), ma quando viene richiesta una lettura coerente, il servizio restituirà una visualizzazione corrente all'elemento o un errore.
- Couchbase offre una gamma di opzioni di coerenza e disponibilità durante una partizione e anche una gamma di opzioni di latenza e di coerenza senza partizione. A differenza della maggior parte degli altri database, Couchbase non ha un singolo set di API né scala / replica tutti i servizi dati in modo omogeneo. Per le scritture, Couchbase favorisce la coerenza rispetto alla disponibilità che lo rende formalmente CP, ma sulla lettura c'è più variabilità controllata dall'utente a seconda della replica dell'indice, del livello di coerenza desiderato e del tipo di accesso (scruve di documenti singola vs scansione dell'intervallo rispetto alla ricerca full-text, ecc.). Inoltre, c'è quindi un'ulteriore variabilità a seconda della replica cross-datacenter (XDCR) che prende più cluster CP e li collega con la replica asincrona e Couchbase Lite che è un database incorporato e crea una topologia distribuita completamente multi-master (con tracciamento delle revisioni).
- Cosmos DB supporta cinque livelli di consistenza sintonibile che consentono compromessi tra C / A durante P e L / C durante E. Cosmos DB non viola mai il livello di consistenza specificato, quindi è formalmente CP.
- MongoDB può essere classificato come sistema PA/EC. Nel caso di base, il sistema garantisce che le leggi e scritture siano coerenti.
- PNUTS è un sistema PC/EL.
- Hazelcast IMDG e la maggior parte delle reti di dati in memoria sono un'implementazione di un sistema PA / CE; Hazelcast può essere configurato come EL piuttosto che EC.[[8] Le primitive di valuta (Lock, AtomicReference, CountDownLatch, ecc.) possono essere sia PC/EC o PA/EC.[[9]
- FaunaDB implementa Calvin, un protocollo di transazione creato dal Dr. Daniel Abadi, l'autore [3]del teorema PACELC, e offre agli utenti controlli regolabili per il compromesso LC. È PC/CE per transazioni rigorosamente serializzabili e EL per le leggi serializzabili.

DDBS	P + A	P + C	E + L	E + C
Aerospike [[10]	✓	Solo pagato	Facoltativo	✓
Bigtable / HBase		✓		✓
Galleria di Cassandra	✓		✓[a]	✓[a]
Il Cosmo DB		✓	✓- [b]	
Base di divano	✓		✓	✓
Dinamo	✓		✓[a]	
DinamoDB di DynamoDB		✓	✓	✓
FaunaDB [[12]		✓	✓	✓
Hazelcast IMDG[[8][[9]	✓	✓	✓	✓
Il megastore		✓		✓
Il MongoDB	✓			✓
Cluster di MySQL		✓		✓
PNUTS di PNUTS		✓	✓	
Tutti i ristoranti a PostgreSQL	✓	✓	✓	✓
Il Riak	✓		✓[a]	
La SpiceDB [[13]		✓	✓	✓
VoltDB/H-Store		✓		✓

Si veda anche

- Teorema della PAC
- Il modello di coerenza
- Fallacies del calcolo distribuito
- Architettura Lambda (soluzione)
- Paxos (la scienza del computer)
- Triangolo di gestione del progetto
- Raft (algoritmo)
- Il trilemma di :

Le note

a. Dynamo, Cassandra e Riak hanno impostazioni regolabili dall'utente per controllare il tradeoff di LC.[[6]
b. Cosmos DB ha cinque livelli di coerenza selezionabili per controllare il tradeoff LC.[11]

Le referenze

1. Golab, Wojciech (2018). "Proving PACELC". (https://dl.acm.org/doi/10.1145/3197406.3197420) *Notizie di ACM SIGACT*. **49:** 73– 81 doi : 10.1145/3197406.3197420 (https://doi.org/10.1145%2F3197406.3197420). S2CID 3989621 (https://api.semanticscholar.org/CorpusID:3989621).
2. - Abadi, Daniel J. (2010-04-23). DMS Musing: problemi con CAP e il sistema NoSQL poco conosciuto di Yahoo (https://dbmsmusings.blogspot.com/2010/04/problems-with-cap-and-yahoos-little.html). Recuperato 2016-09-11.
3. - Abadi, Daniel J. "Procedi di coerenza nella progettazione del sistema di database distribuiti moderni" (http://www.cs.umd.edu/~abadi/papers/abadi-pacelc.pdf) (PDF). - Università di Yale.
4. Abadi, Daniel (15 luglio 2019). "I pericoli della consistenza condizionale garantiscono". (https://dbmsmusings.blogspot.com/2019/07/the-dangers-of-conditional-consistency.html) *DBMS Musings*. Recuperato il 29 agosto 2024.
5. *Progettare applicazioni ad alta intensità di dati: le grandi idee dietro sistemi affidabili, scalabili e di mantenimento*. O'Reilly Media. Il ISBN 978-1449373320- Si'.
6. Abadi, Daniel J.; Murdopo, Arinto (2012-04-17). "Procedi di coerenza nella moderna progettazione del sistema di database (https://www.slideshare.net/arinto/consistency-tradeoffin moderndb) distribuiti". Recuperato 2022-07-18.
7. "Tabell globali - replica multi-Regione per (https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GlobalTables.html) DynamoDB". *Documentazione di AWS*. Recuperato il 4 gennaio 2023.
8. Abadi, Daniel (2017-10-08). DMS Musing: Hazelcast e il sistema MIFico PA/EC (https://dbmsmusings.blogspot.com/2017/10/hazelcast-and-mythical-paec-system.html). *DBMS Musings*. Recuperato 2017-10-20.
9. "Hazelcast IMDG Manuale (https://docs.hazelcast.org/docs/4.0.2/manual/html-single/index.html#cp-subsystem) di riferimento". (https://docs.hazelcast.org/docs/4.0.2/manual/html-single/index.html#cp-subsystem) *docs.hazelcast.org*. Recuperato 2020-09-17.
10. Porter, Kevin (29 marzo 2023). "Dove cade l'aerospike in PACELC?" (https://discuss.aerospike.com/t/where-does-aerospike-fall-in-pacelc/10111/2). *Forum della comunità Aerospike*. Recuperato il 30 marzo 2023.
11. Livelli di coerenza in Azure Cosmos DB (https://learn.microsoft.com/en-us/azure/cosmos-db/consistency-levels). Recuperato 2021-06-21.
12. Abadi, Daniel (2018-09-21). DMS Musings: i sistemi di database NewSQL non riescono a garantire la coerenza e io incolpo Spanner (https://dbmsmusings.blogspot.com/2018/09/newsql-database-systems-are-failing-to.html). *DBMS Musings*. Recuperato 2019-02-23.
13. Zelinskie, Jimmy (2024-04-23). Concetti di SpetteDB: Coerenza (https://authzed.com/docs/spicedb/concepts/consistency). *La documentazione di SpiceDB*. Recupero 2024-05-02.

Collegamenti esterni

- "I compromessi di coerenza nella moderna progettazione di database distribuiti", di Daniel J. Abadi, (https://www.cs.umd.edu/~abadi/papers/abadi-pacelc.pdf) documento originale della Yale University (https://www.cs.umd.edu/~abadi/papers/abadi-pacelc.pdf) che ha formalizzato PACELC
- "Problemi con CAP, e il sistema NoSQL poco conosciuto di Yahoo", di Daniel J. Abadi, Università di Yale (https://dbmsmusings.blogspot.com/2010/04/problems-with-cap-and-yahoos-little.html). Post sul blog originale che per primo ha descritto PACELC
- "Proving PACELC", di Wojciech Golab, Università di Waterloo (https://uwaterloo.ca/distributed-algorithms-systems-lab/sites/default/files/uploads/files/proving_pacelc.pdf) Prova formale del

teorema PACELC

Espedito da " https://en.wikipedia.org/w/index.php?title-PACELC?theorem&oldid?1261384014 "