Introduction aux bases de données NoSQL

Houcine Senoussi

October 17, 2017

Introduction

- Les bases de données NoSQL ont en commun les caractéristiques suivantes :
  - Elles ne sont pas basées sur le **modèle relationnel**. N'exigent pas un schéma (structure) pré-défini.
  - Elles sont **distribuées** : elles tirent profit de la possibilité de bâtir de larges clusters de serveurs reliés par des réseaux rapides.
    - ce qui implique des données **réparties** et **répliquées**, et des traitements **parallèles**.
  - Les données que ces bases de données permettent de stocker, traiter et analyser sont volumineuses, variées et d'une grande vélocité : les BDDs NoSQL sont fortement liées au Big Data.

Introduction

- Historiquement les premières bases NoSQL sont Bigtable et Dynamo.
  - BigTable (2004-2006): créée par les ingénieurs de Google pour les besoins de leur société (notamment indexation du Web, Google Analytics et GooglEarth). Ses concepteurs la définissent comme un système de 'stockage **distribué** conçu pour évoluer vers une très grande taille (**scalability**) : plusieurs **petabytes** de données **réparties** sur plusieurs **milliers de serveurs**. L'article (2006) souligne le fait que Bigtable avait atteint 4 objectifs essentiels : large utilisabilité (applications diverses et nombreuses), **scalabilité**, haute performance et haute disponibilité des données.

Introduction

- Bigtable (suite)
  - Le modèle de données de Bigtable est défini comme suit : chaque donnée (on dit une cellule) est définie par un triplet (*ligne*, *colonne*, *timestamp*). Les données sont regroupées dans des 'ColumnFamily' mais chaque ligne n'a pas forcément le même nombre de colonne. Nous pouvons avoir plusieurs versions de la même donnée dans la même Columnfamily, indexées par le Timestamp. Un mécanisme de grabage-collector permet de 'faire le ménage' régulièrement pour ne garder que les versions les plus récentes.

Introduction

- Dynamo (Amazon 2007)
  - définie comme un 'highly available key-value storage system'. Pour atteindre un aussi haut niveau de disponibilité les ingénieurs ont dû sacrifier la cohérence (consistency) dans certaines situations. Objectifs : performance, fiabilité (reliability), et scalabilité. L'accent est mis sur la fiabilité car son affaiblissemnt peut avoir des conséquences néfastes sur le business.
  - L'accent est mis sur la disponibilité (Availability) définie comme suit : customers should be able to view and add items to their shopping cart even if disks are failing, network routes are flapping, or data centers are being destroyed by tornados. Il s'agit donc en même temps de disponibilité et de tolérance aux pannes.

Introduction

- Dynamo (suite)
  - L'accent est aussi mis sur le coté hautement **décentralisé** de Dynamo : on peut ajouter ou supprimer des noeuds sans modifier le fonctionnement de l'ensemble.

Types de BDDs NoSQL

- Même si elles ont beaucoup évolué depuis leur naissance, les BDDs NoSQL peuvent être rangées dans deux catégories principales :
  - Les BDDs graphe : Les données et leurs relations forment des graphes (donnée = sommet, relation = arête). Neo4J est l'exemple le plus connu de ces bases.
  - Les BDDs clé-valeur : Ces bases de données n'ont pas de 'schéma' (structure comme dans le modèle relationnel) pré-défini. Les données sont stockées sous la forme de couples *(clé unique, valeur correspondante)*. Étant donné que, contrairement aux BDDRs ces BDDs n'intègrent pas de contrainte autre que l'unicité de la clé, toute la complexité des relations entre les données est gérée par l'application et pas au niveau de la base.

Types de BDDs NoSQL

- Les BDDs Clé-Valeur peuvent être réparties en plusieurs sous-familles :
  - Clé-Valeur simple : Dynamo par exemple.
  - Colonne : Les données sont définies par des lignes et des colonnes, regroupés en familles de colonnes ce qui les fait ressembler aux tables du modèle relationnel mais ici les ligne n'ont pas forcément les mêmes colonnes. Exemples : Bigtable et Cassandra.
  - Document : les couples clés-valeurs sont regroupés en blocs. Chaque bloc est un document. Les documents à leur tour peuvent être regroupés en collections (ce qui correspond aux tables du modèle relationnel). Exemple : CouchDB et MongoDB.

Autres Propriétés des BDDS NoSQL

- Distribuée : Données réparties sur plusieurs serveurs/traitements parallèles.
- Décentralisée : les serveurs jouent le même rôle. pas de relation maître-esclave.
- Scalable : on peut ajouter de nouveaux serveurs sans modifier l'organisation ni dégrader les performances.
- Availability/Consistency/fault Tolerance : voir plus bas.

De quoi s'agit-il ?

- Consistency : Tous les clients accédant à une même donnée lisent la même valeur.
- Availability : Tous les clients ont un accès immédiat à une donnée qu'ils souhaitent consulter (lecture) ou modifier (écriture).
- Partition tolerance : la base de données répartie sur plusieurs machines continue à fonctionner en cas d'un problème réseau (machine inaccessible).

De quoi s'agit-il ?

En résumé :

Lorsque l'on considère la cohérence des données (**consistency**), leur disponibilité (**availability**), et la tolérance aux pannes (**partition tolerance**), seuls deux de ces trois exigences peuvent être assurées en même temps.

De quoi s'agit-il ?

- Les données existent en plusieurs copies (réplication).
  - Pourquoi ? à la fois pour en assurer l'accès à un plus grand nombre et pour assurer la tolérance aux pannes (une copie inaccessible suite à une panne $\rightarrow$ accéder aux autres copies).
- Pour obtenir une cohérence stricte toutes les opérations de mise à jour doivent avoir lieu de manière synchrone : les unes à la suite des autres. Ce qui signifie qu'une "transaction" doit **bloquer/verrouiller** l'accès à **toutes les copies** des données qu'elle est en train de mettre à jour jusqu'à la fin de l'opération. Cela forcerait les autres 'transactions' concurrentes à attendre.
  - $\rightarrow$ Réduction de la disponibilité.

Théorème de Brewer

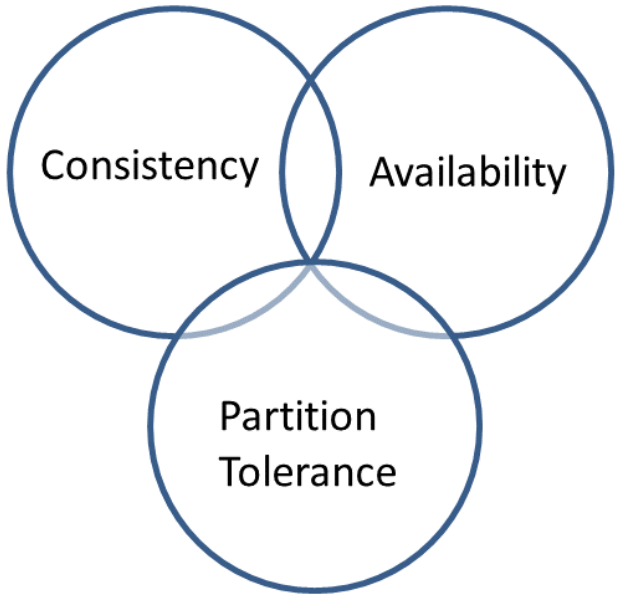

Figure: Relations entre C-A-P

Théorème de Brewer

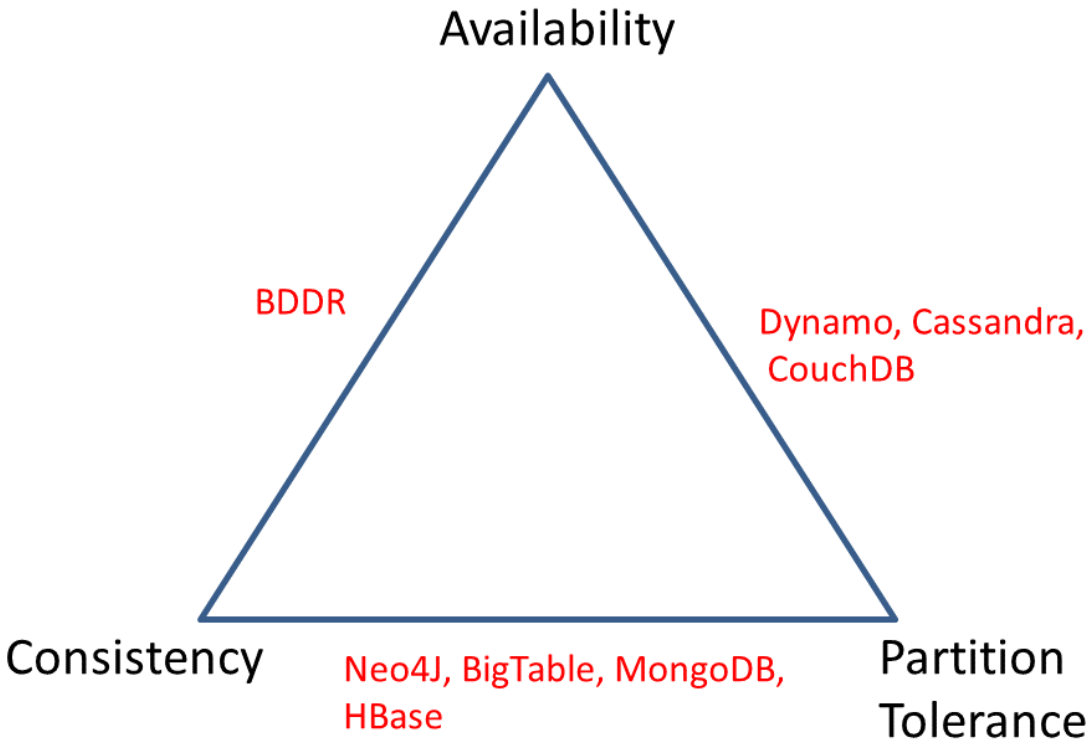

Figure: Repartitions des BDDs selon C-A-P

Théorème de Brewer

- Un système distribué ne peut se passer de partition tolerance.
  - Il reste donc à choisir entre les deux autres propriétés.
- Il est intéressant de noter que le choix de cette propriété, c'est-à-dire entre le cotés CP et AP de la figure, n'est pas directement lié à la catégorie de la base de donnée :
  - Une BDD graphe (Neo4J) et une BDD document appartiennent à la famille CP.
  - Deux BDD Document (CouchDB et MongoDB) appartiennent à deux choix différents.

Exemple : Cassandra

- Dans le chapitre suivant nous étudions plus particulièrement la base de données *Cassandra*.

Conclusion

- Caractéristiques et familles des bases de données NoSQL.
- Suite : nous allons illustrer le fonctionnement des bases de données NoSQL par un exemple : Cassandra.

Conclusion

- Managing Big data : SQL or NoSQL ? IDC 2012.
- Cassandra, the definitive guide. Eben Hewil. O'reilly, 2011.