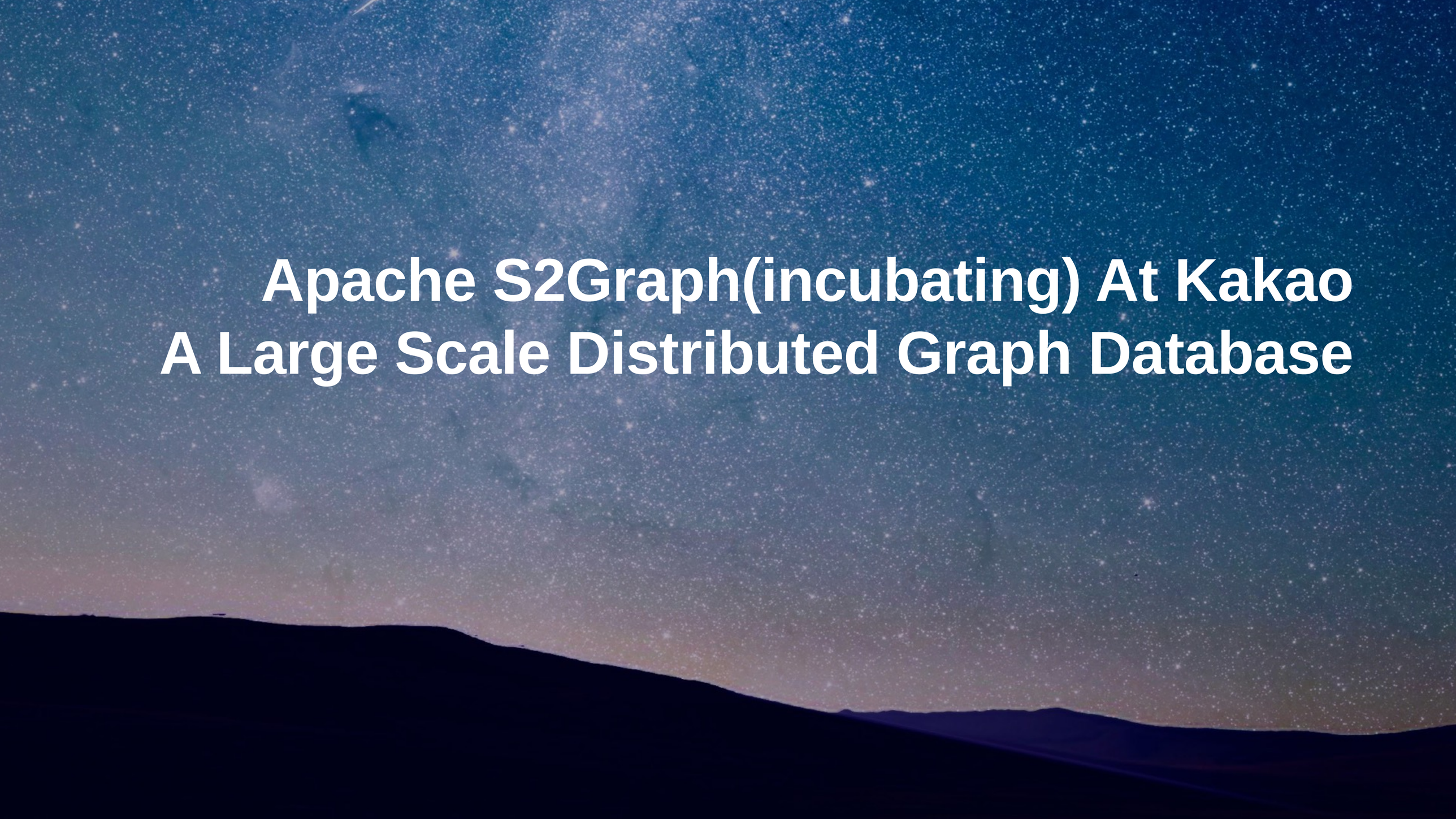

Apache S2Graph(incubating) At Kakao
A Large Scale Distributed Graph Database

TOC

kakao

1. Technical Challenges.
2. What is S2Graph.
3. Use Cases.
4. Real World Use Cases.

Technical Challenges

Why We Got into Graph Databases

kakao

+30 services at Kakao are powered by S2Graph

Graph Databases

- Relationship-Oriented
- Expressive yet Simple

Our Data

- Highly Connected
- Complex Relations

Social Graph?

kakao

Relations + Activities = Social Graph

- 

is friend of 

- 

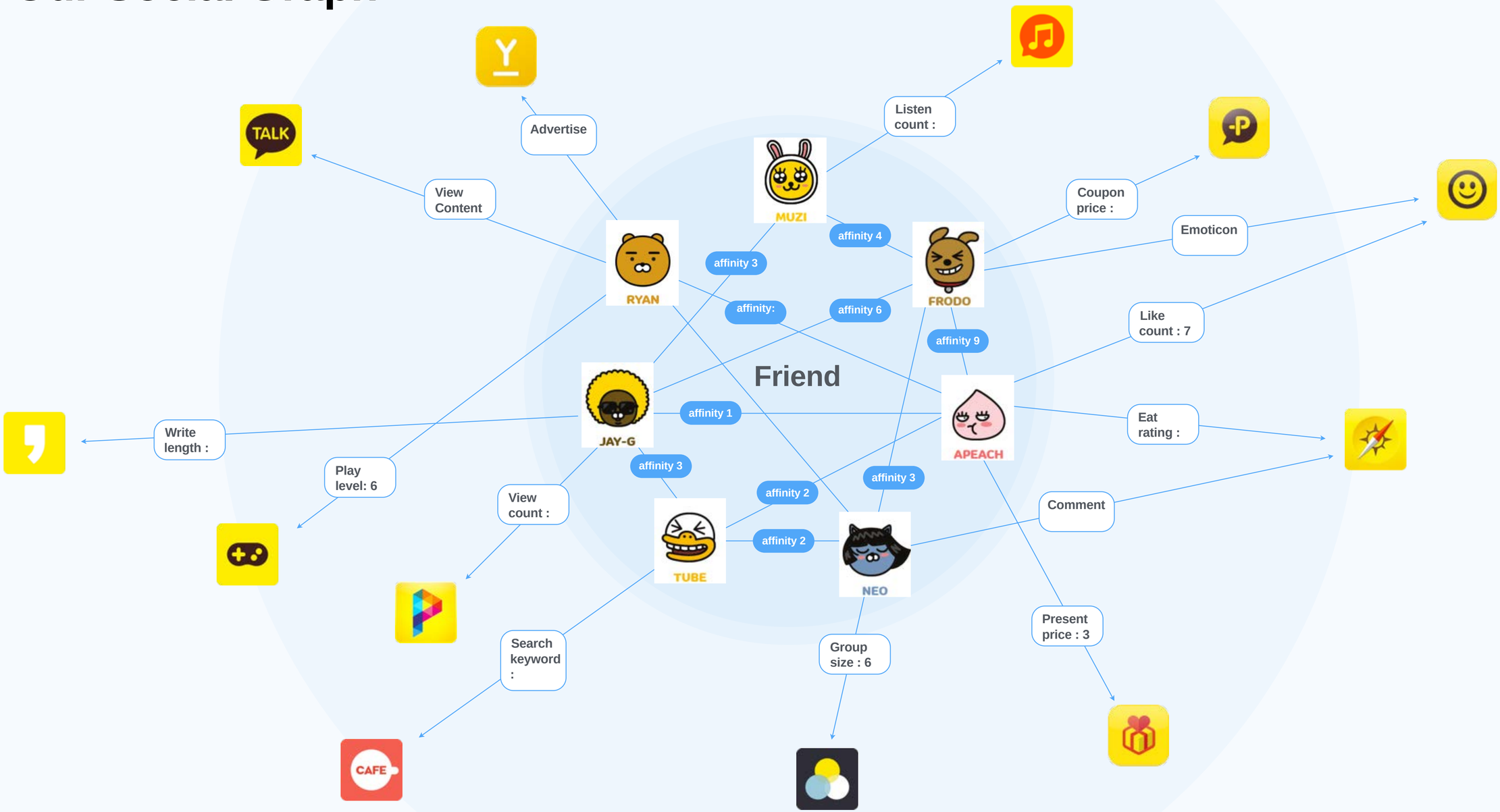
listen to “Hello” by Adele.

- 

is playing KakaoGame 

Our Social Graph

kakao



Technical Challenges

kakao

1.Large graph constantly changing.

1. Total # of Edge: + 1 trillion and growing.
2. Social Network: more than 10 billion edges, 200 million vertices, 50 million realtime update on social network.
3. User activities(Click, Like, Share, Buy): 2.5 ~ 3 billion real-time incoming edges, 50 billion batch processed edges.

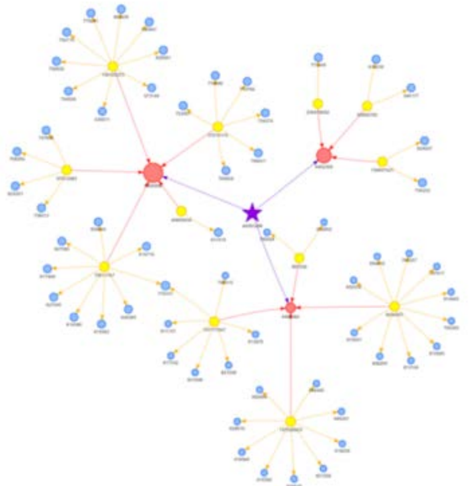
2. Low latency for breadth first search traversal on connected data.

1. Peak graph-traversing query per minute: 4 million
2. Average response time: 50 ms

3. Update should be applied into query result in real-time.

4. Support for Dynamic Ranking logic

1. push strategy: hard to change data ranking logic dynamically.
2. pull strategy: can try various data ranking logic



Why Did We Build It?

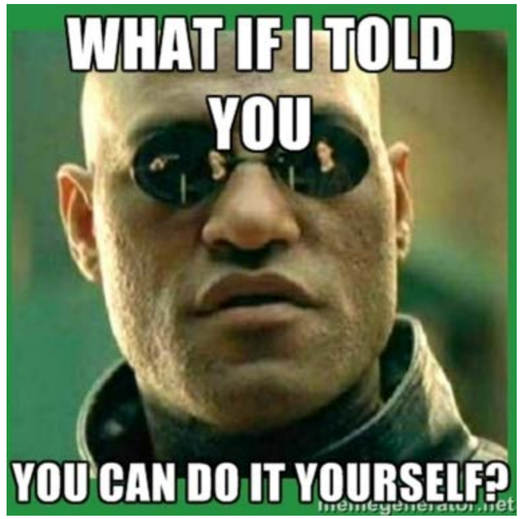
kakao

Existing solutions weren’t performant enough for our needs.

Especially,

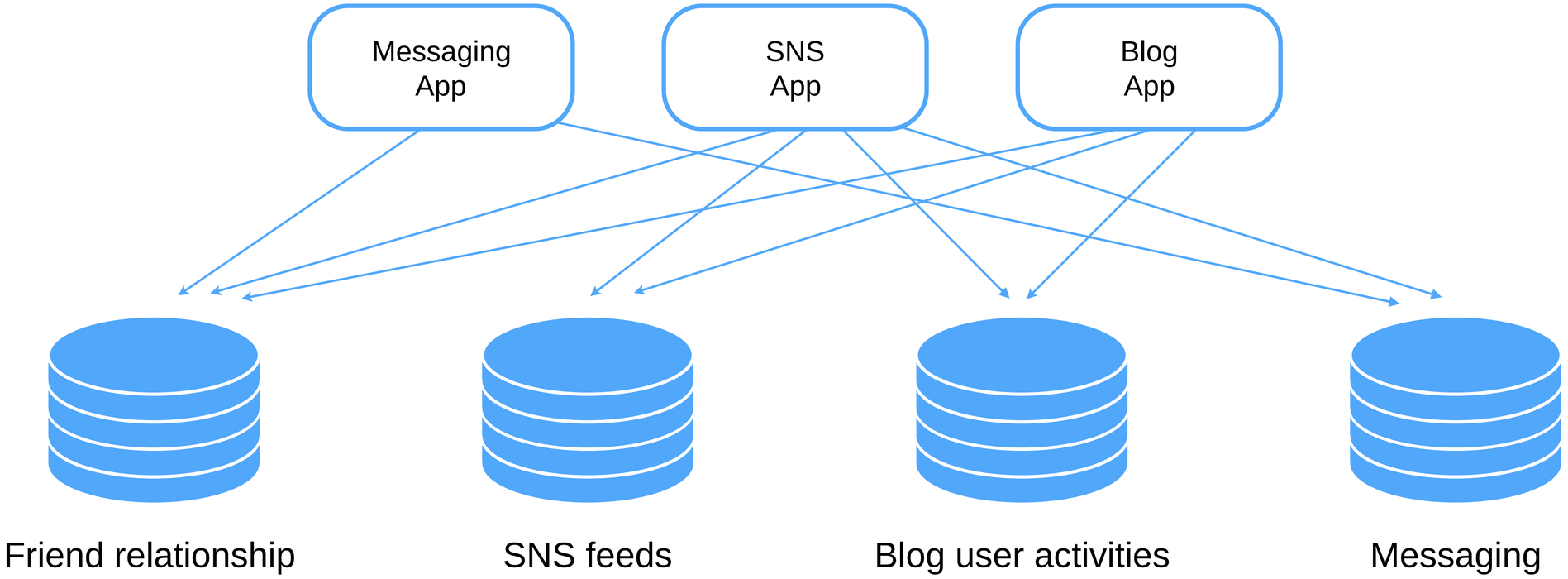
1. Maintaining a mutable graph at scale was not supported. (i.e. Updates/ Deletes were slow!)
2. Breadth First Search traversal was not fast enough.

So we built our own!



Before

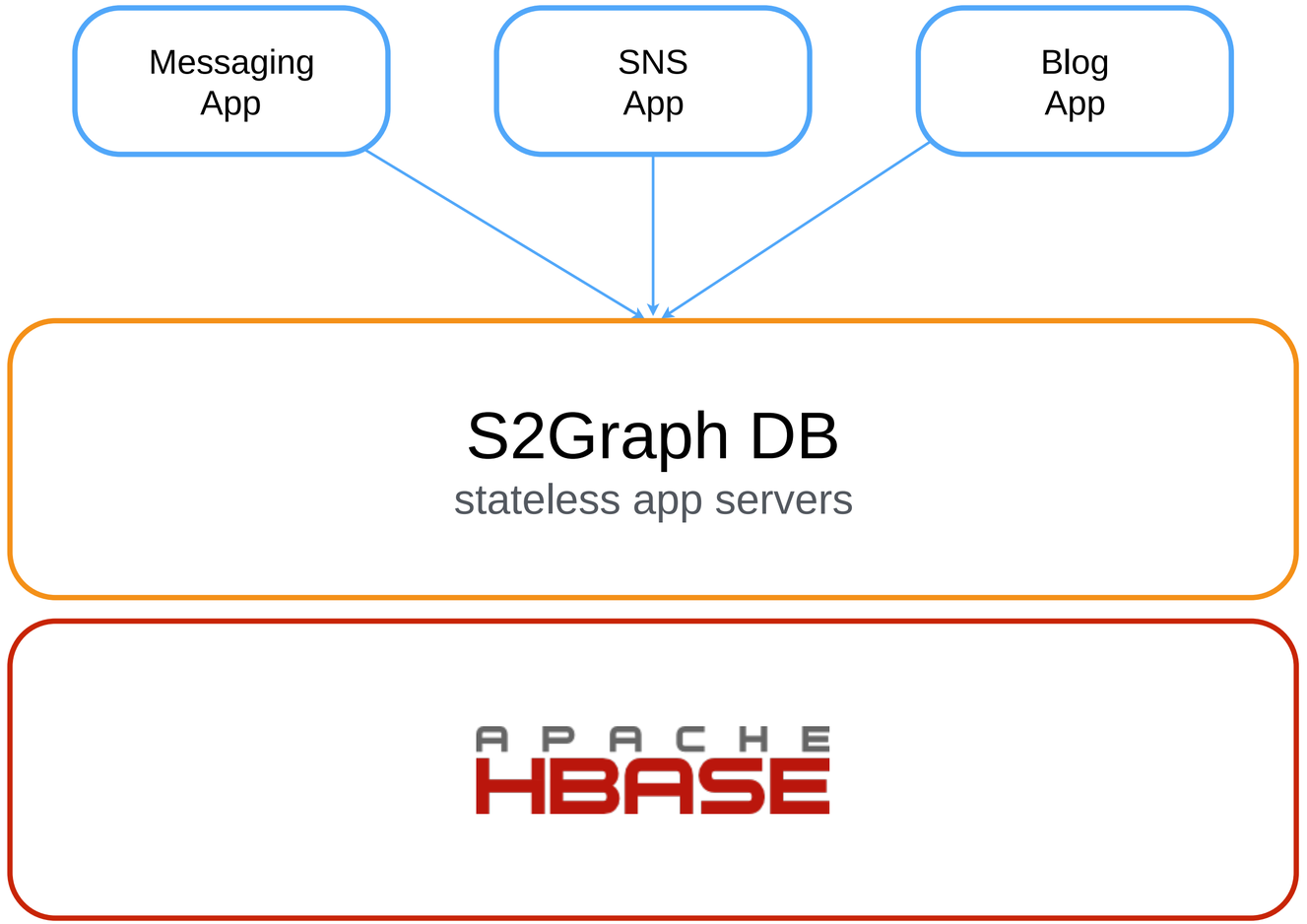
kakao



- Each app server should know each DB’s sharding logic.
- Highly inter-connected architecture

After

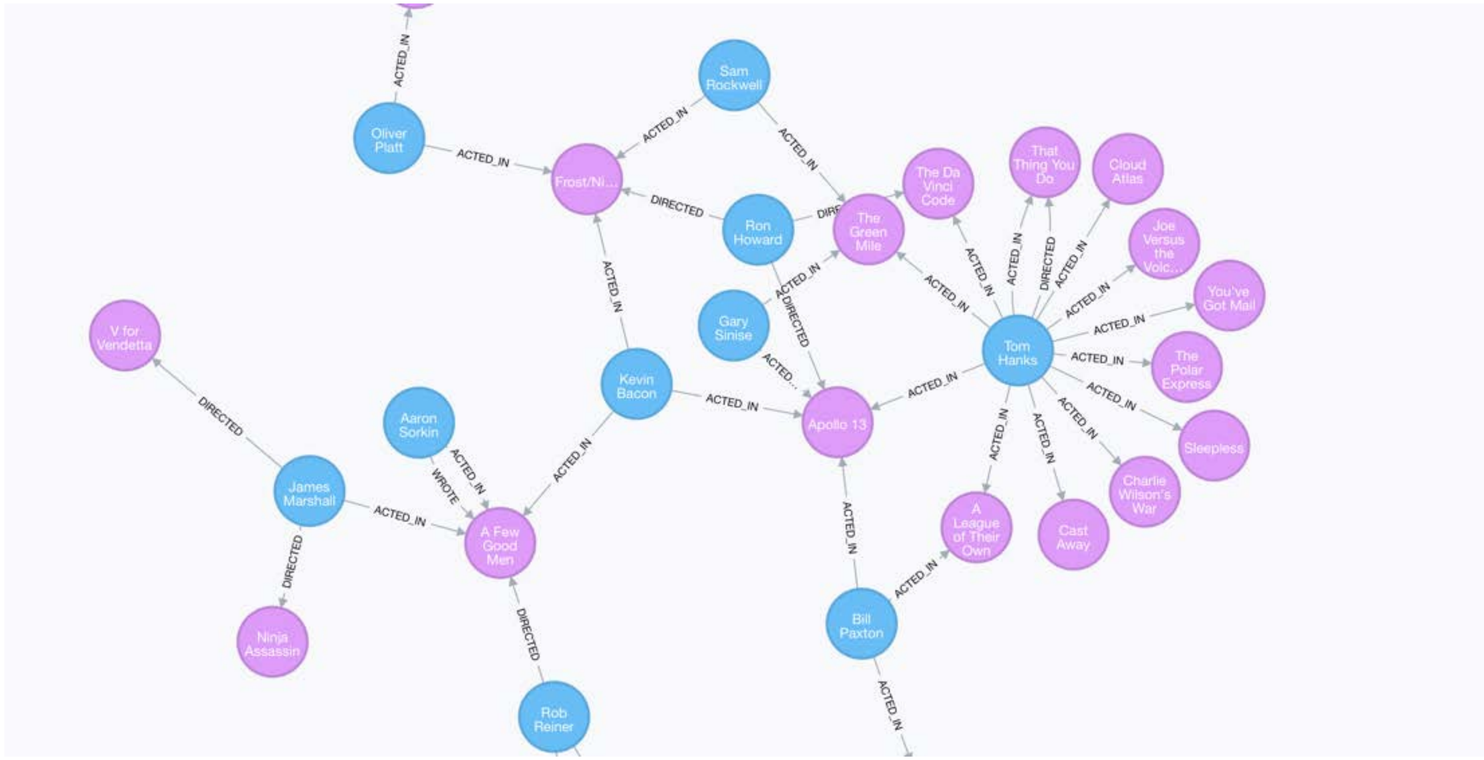
kakao


Messaging App
SNS App
Blog App
S2Graph DB
stateless app servers
APACHE HBASE

What is S2Graph?

What is S2Graph?

kakao

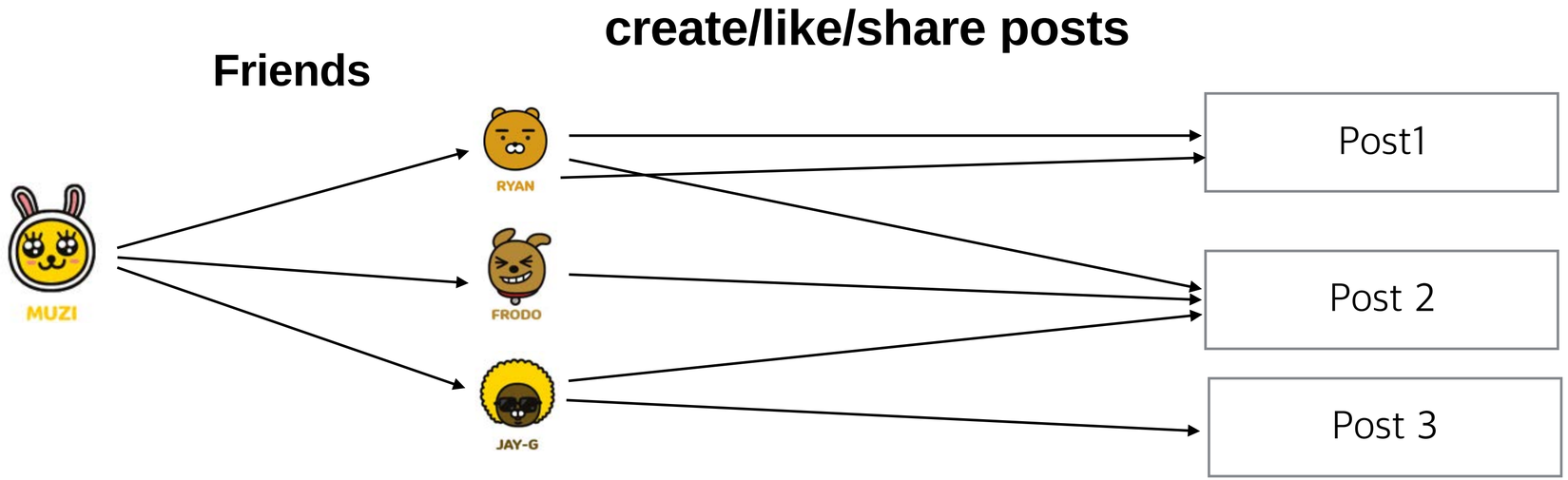


A property graph of actor Kevin Bacon
(http://propellerhead.ca/2016/02/fun-with-graph-databases)

- Property Graph Model: Vertices + Edges + Properties
- S2Graph = Property Graph Model + Scalability + Fast CRUD Operations
- Graph-processing layer atop HBase

Example: News Feed (cont)

kakao



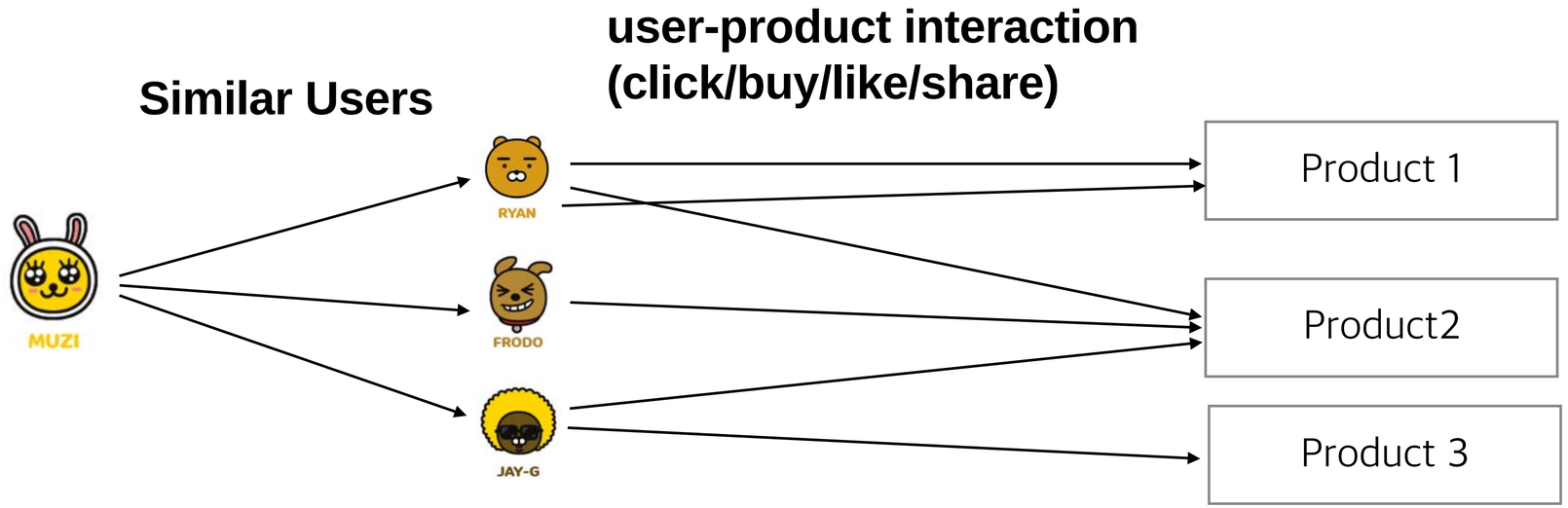
Posts that my friends interacted.

```
SELECT a.*, b.*
FROM    friends a, user_posts b
WHERE a.user_id = b.user_id WHERE b.updated_at >= yesterday
and b.action_type in (‘create’, ‘like’, ‘share’)
```

```
curl –XPOST localhost:9000/graphs/getEdges –H 'Content-Type: Application/json' –d '
{
    "srcVertices": [{"serviceName": "s2graph", "columnName": “user_id", "id":1}],
    "steps": [
      [{"label": "friends", "direction": "out", "limit": 100}], // step
      [{"label": “user_posts", "direction": "out", "limit": 10, “where”: “created_at >= yesterday”}]
    ]
}
'
```

Example: Recommendation(User-based CF) (cont)

kakao



Products that similar user interact recently.

```
SELECT a.* , b.*
FROM  similar_users a, user_products b
WHERE a.sim_user_id = b.user_id
AND  b.updated_at >= yesterday
```

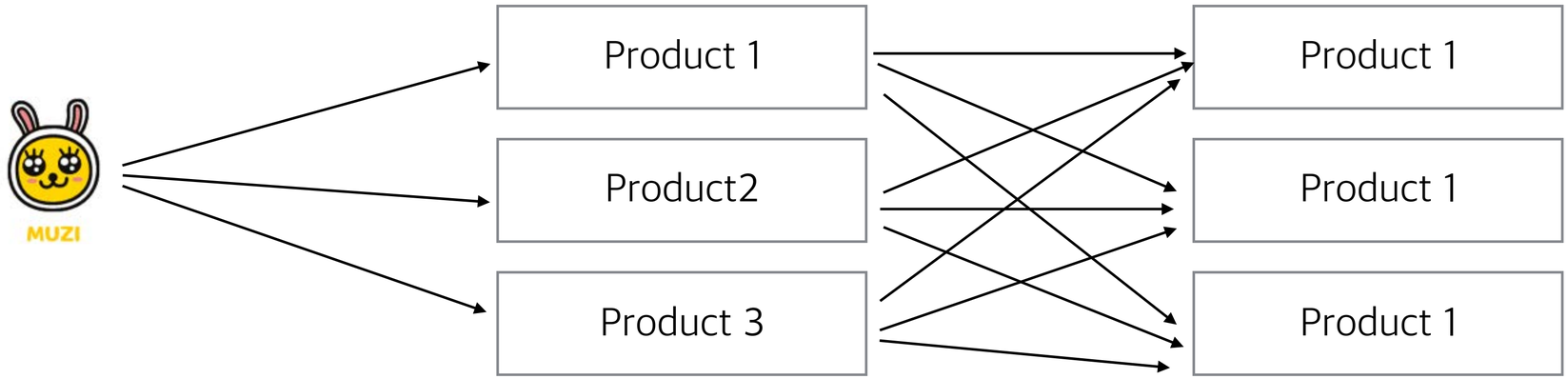
```
curl -XPOST localhost:9000/graphs/getEdges -H 'Content-Type: Application/json' -d '
{
     “filterOut”: {“srcVertices”: [{“serviceName”: “s2graph”, “columnName”: “user_id”, “id”: 1}],
        “steps”: [[{“label”: “user_products_interact”}]]
      },
    "srcVertices": [{"serviceName": "s2graph", "columnName": “user_id”, "id":1}],
    "steps": [
      [{"label": “similar_users", "direction": "out", "limit": 100, “where”: “similarity > 0.2”}], // step
      [{"label": “user_products_interact”, "direction": "out", "limit": 10,
        “where”: “created_at >= yesterday and price >= 1000”}]
    ]
}
'
```

Example: Recommendation(Item-based CF) (cont)

kakao

user-product interaction (click/buy/like/share)

Similar Products



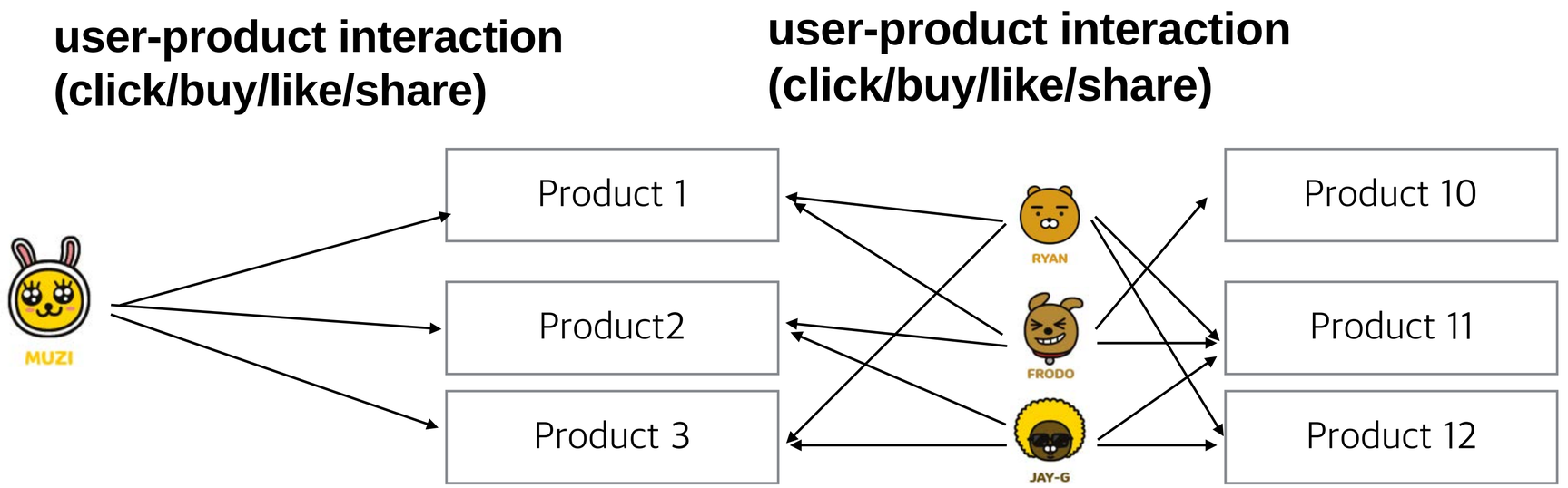
Products that are similar to what I have interested.

```
SELECT a.* , b.*
FROM  similar_ a, user_products b
WHERE a.sim_user_id = b.user_id
AND  b.updated_at >= yesterday
```

```
curl -XPOST localhost:9000/graphs/getEdges -H 'Content-Type: Application/json' -d '
{
    "srcVertices": [{"serviceName": "s2graph", "columnName": “user_id", "id":1}],
    "steps": [
      [{"label": “user_products_interact", "direction": "out", "limit": 100,
        “where”: “created_at >= yesterday and price >= 1000”}],
      [{"label": “similar_products”, "direction": "out", "limit": 10, “where”: “similarity > 0.2”}]
    ]
}
'
```

Example: Recommendation(Spreading Activation) (cont)

kakao



Products that is interacted by users who interacted on products that I interact

```
SELECT b.product_id, count(*)
FROM  user_products a, user_products b
WHERE a.user_id = 1
AND a.product_id = b.product_id
GROUP BY b.product_id
```

```
curl -XPOST localhost:9000/graphs/getEdges -H 'Content-Type: Application/json' -d '
{
    "srcVertices": [{"serviceName": "s2graph", "columnName": “user_id", "id":1}],
    "steps": [
      [{"label": “user_products_interact", "direction": "out", "limit": 100, “where”: “created_at >= yesterday and price >= 1000”}],
       [{"label": “user_products_interact", "direction": "in", "limit": 10, “where”: “created_at >= today”}],
       [{"label": “user_products_interact", "direction": "out", "limit": 10, “where”: “created_at >= 1 hour ago”}],
    ]
}
'
```

Use Cases

1. Storage for user activities and relationships

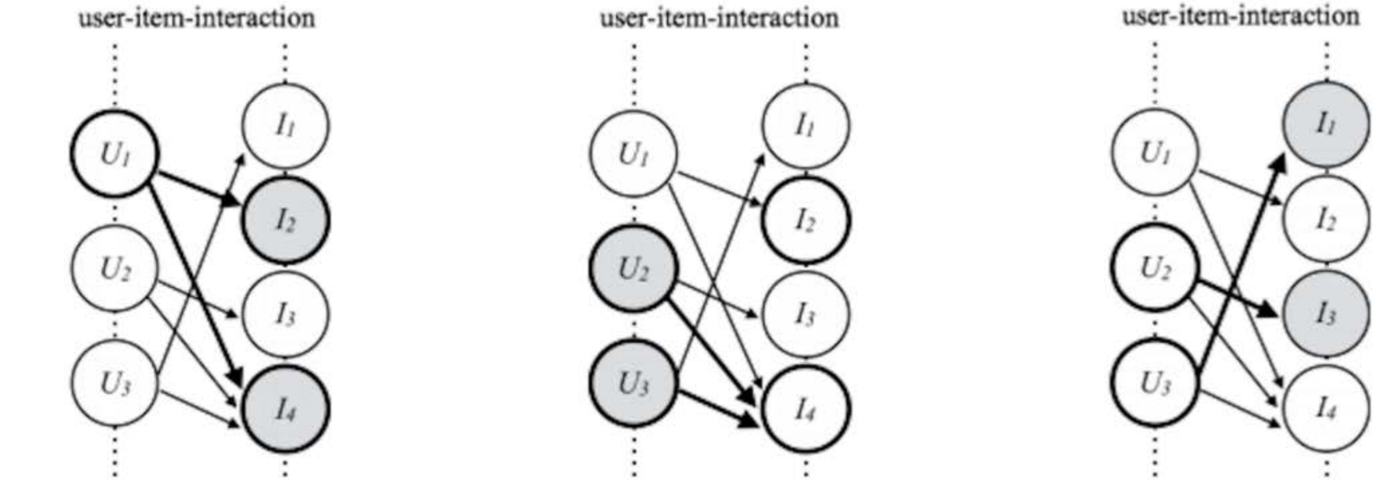
kakao

Friends, clicks, purchases, likes, shares, comments, etc.

2. Real-time Recommendation (Spreading Activation)

kakao

1. Find items a user has reacted to (clicked, purchased..)
2. Find other users who reacted to the same items.
3. Find other items that those users reacted to.



(a) Items liked by the target user (b) users who liked the items (c) items liked by the users

Figure 5: Real-time collaborative filtering by Spreading Activation

2. Real-time Recommendation (cont)

kakao

Composite multiple queries via weighted sum

EX) Ensemble of recommendation algorithms

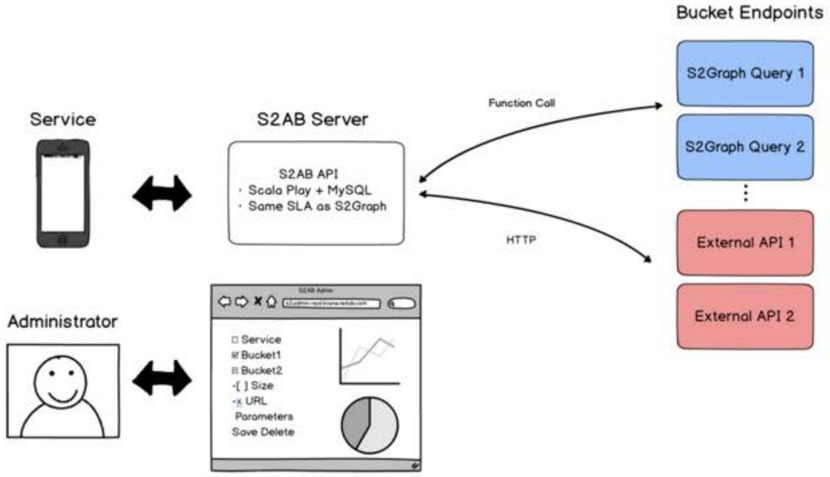
1.Item-based collaborative filtering (CF)
2.User-based CF
3.Matrix Factorization
4.Demographical MP items (for cold-start problem)

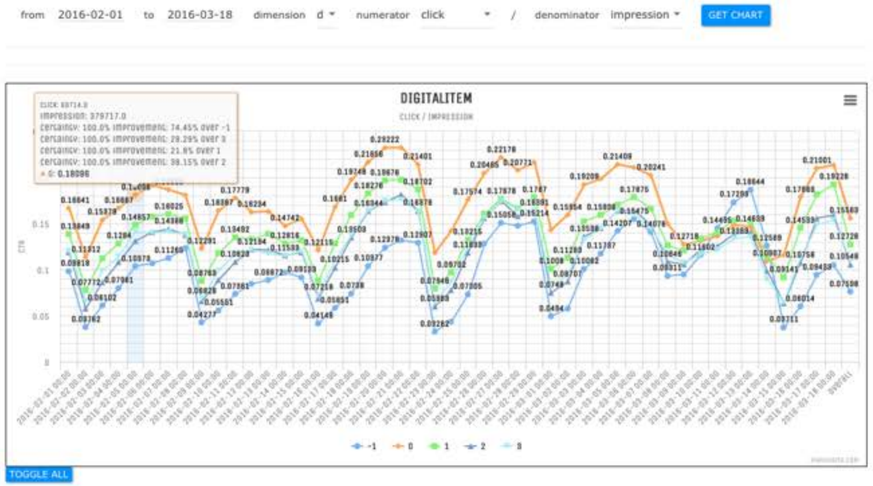
```
{
    "weights": [
        1000,
        100,
        10
        1
    ],
    "queries": [
        {
            "srcVertices": [...], // some user
            "steps": [[{"label": "item_based_cf_label"}]]
        },
        {
            "srcVertices": [...], // some user
            "steps": [[{"label": "user_based_cf_label"}]]
        },
        {
            "srcVertices": [...], // some user
            "steps": [[{"label": "matrix_factorization_label"}]]
        },
        {
            "srcVertices": [...], // some user
            "steps": [
              [{"label": "user_demography_label"}],
              [{"label": "demographical_top_k_label"}]
            ]
        }
    ]
}
```

3. Native A/B (Bucket) Testing

kakao

1.Bucket = Query
2.Track performance of each bucket realtime: CTR, conversion rate..
3.Maximize performance.





Real World Use Cases

Experiment example

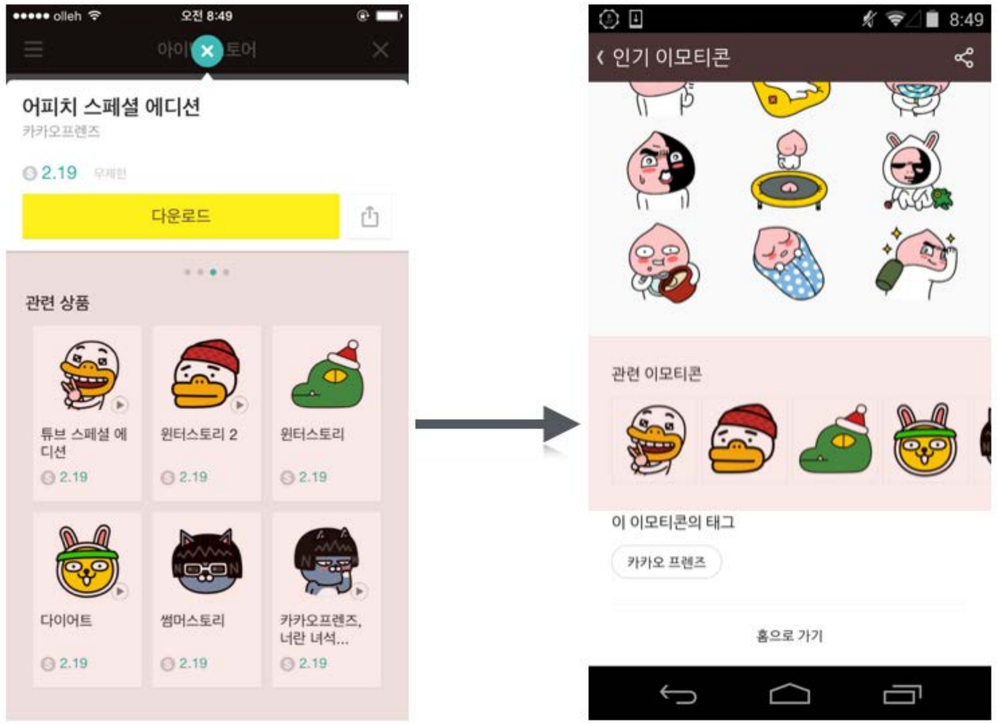
kakao

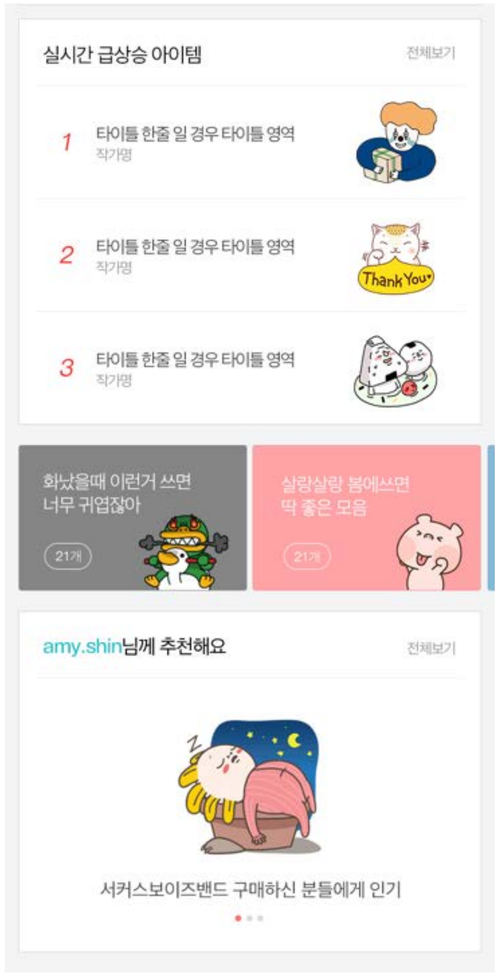
1. User Based Collaborative Filtering
2. Item Based Collaborative Filtering.
3. Matrix Factorization.
4. Content Based Collaborative Filtering.
5. Most Popular.
6. Segmented Most Popular.
7. Spreading Activation.
8. Social Recommendation.

Weighted sum of
each algorithms are also possible.

1. Emoticon Store

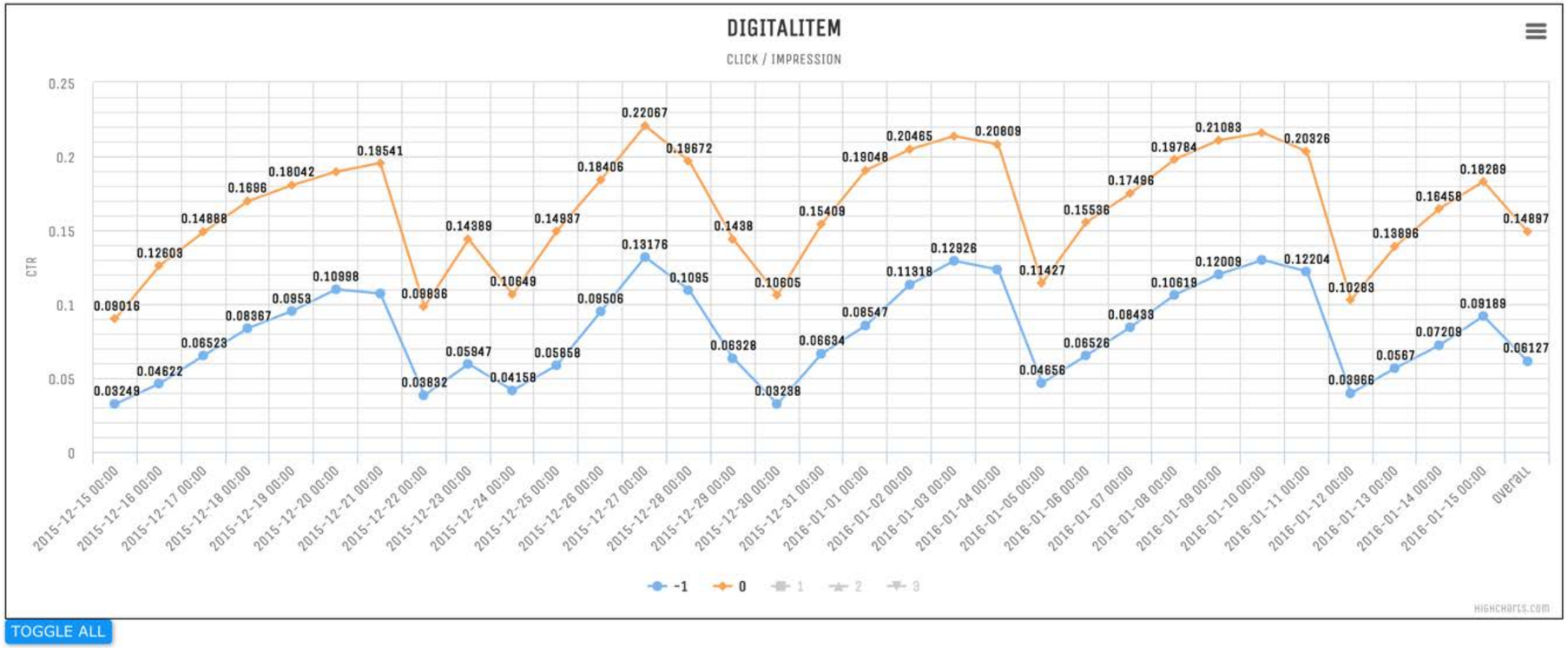
kakao





1. Emoticon Store (cont)

kakao



Baseline(most popular) vs S2Graph

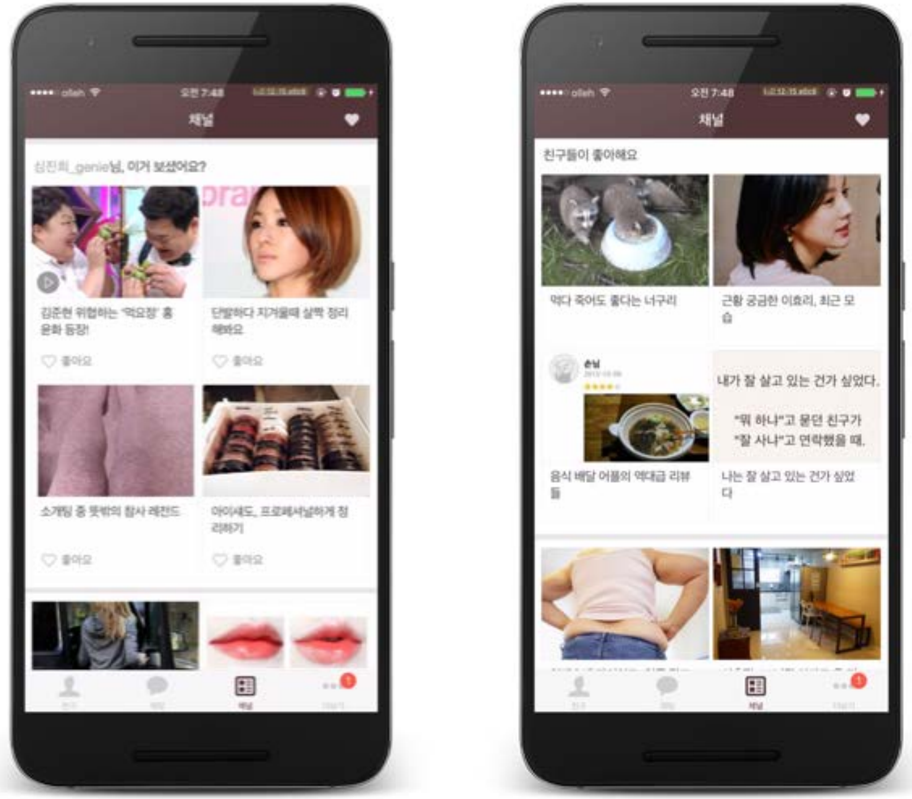
CTR: +137.52%
Purchase/Click: +70.8%
Purchase/Impression: +304%

Before -> After

Total Item Click: x3.8
Unique User with Item Click: x3.5

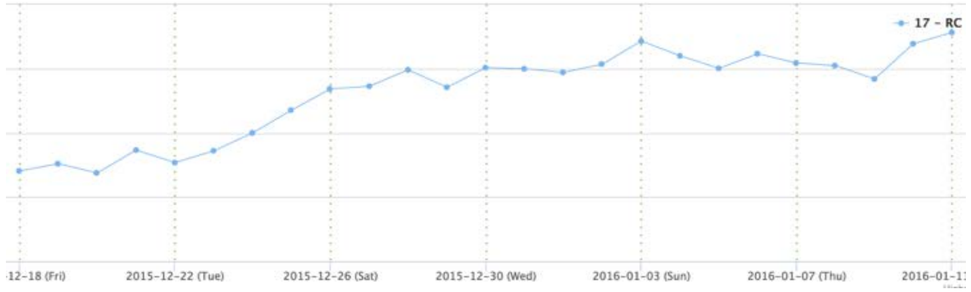
2. Kakao Talk Channel: Recommend Card, Social Card

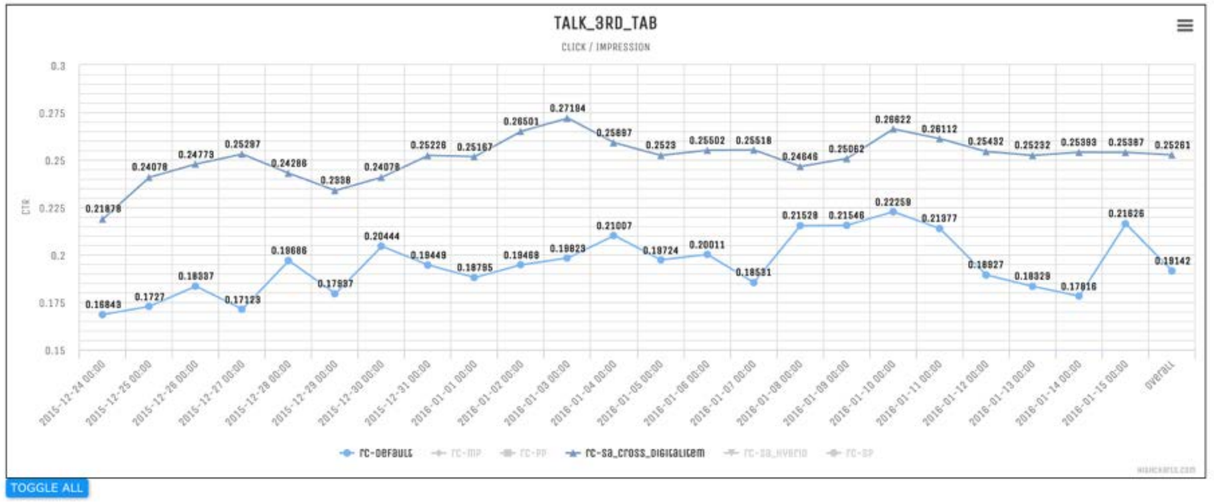
kakao



2. Kakao Talk Channel: Recommend Card (cont)

kakao



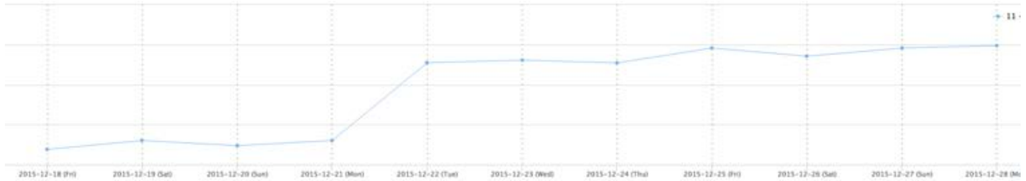


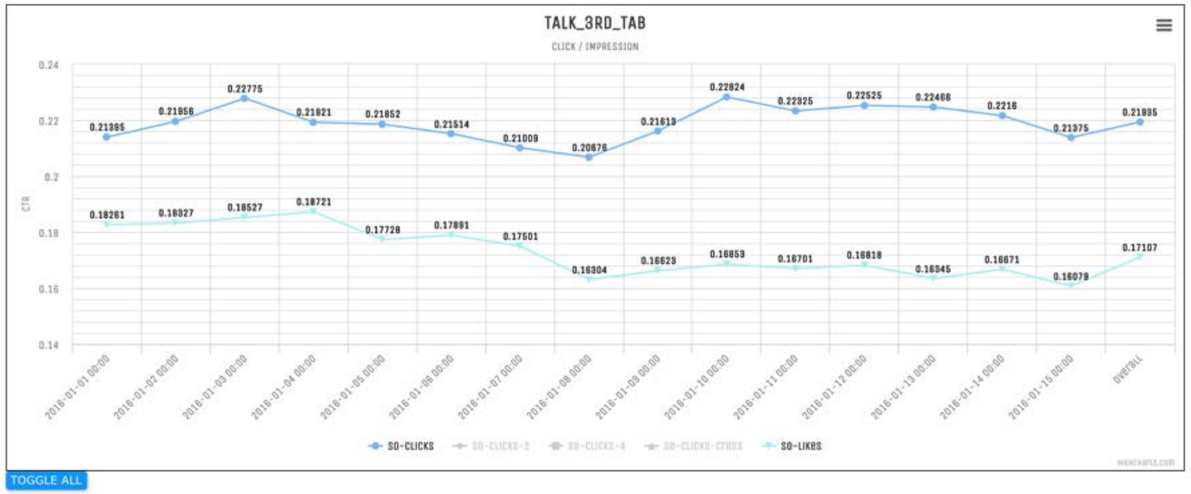
Baseline(most popular) vs S2Graph

Total Click: +123%
CTR: +36.6%
Unique # of contents with clicks: x20

2. Kakao Talk Channel: Social Card (cont)

kakao



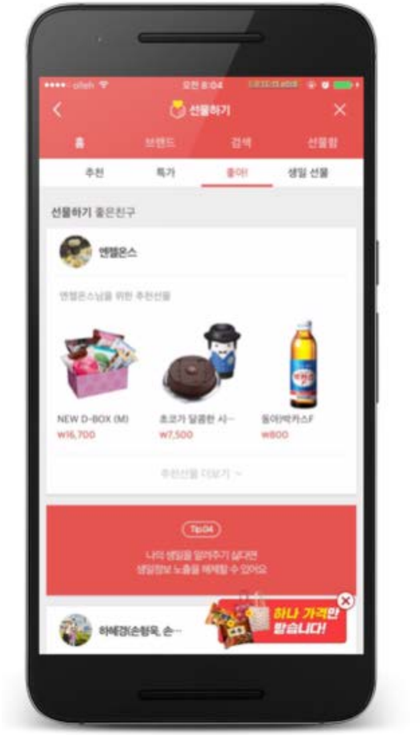


Baseline vs S2Graph

Total Click: +591%
CTR: +23.65%

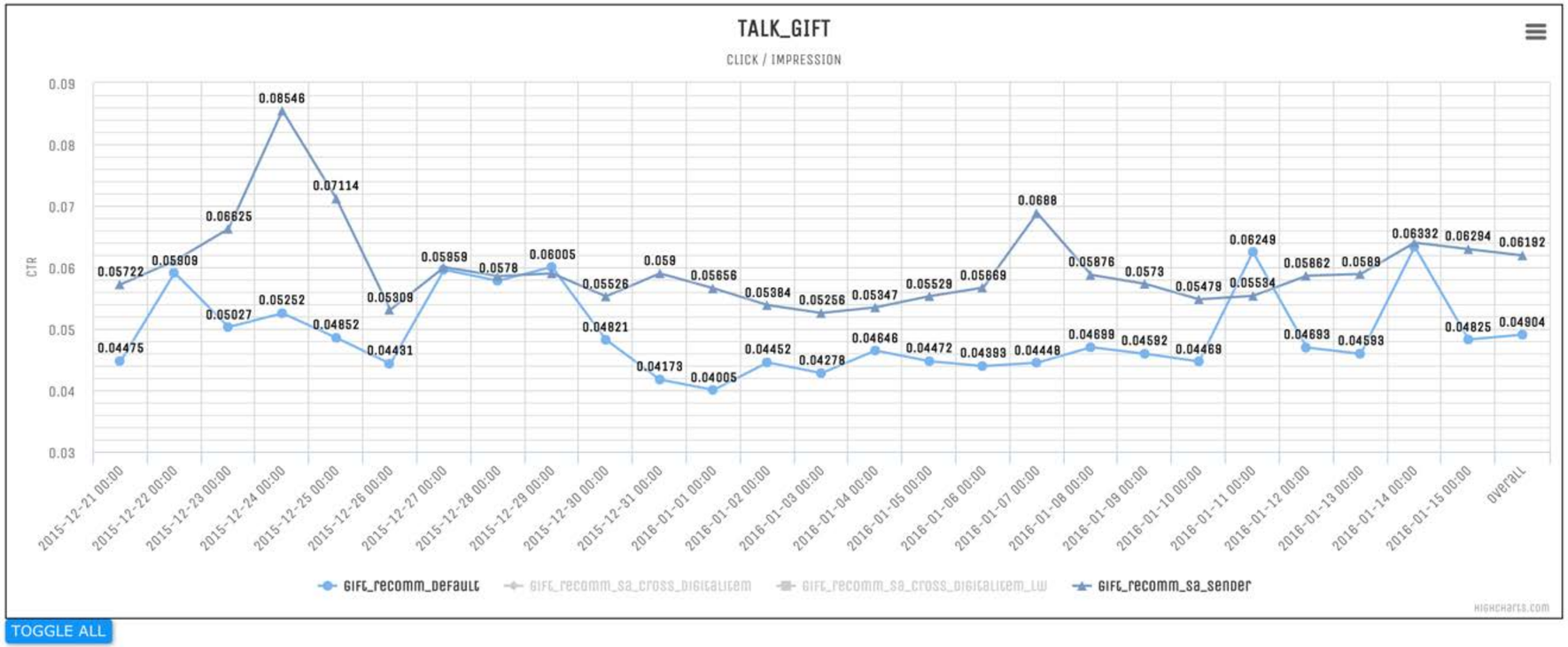
3. Kakao Talk Gift Shop

kakao



3. Kakao Talk Gift Shop (cont)

kakao



Baseline(most popular) vs S2Graph

CTR: +26.03%
Purchase/Click: +344%
Purchase/Impression: +459%

Powered By S2Graph

kakao

Social Platform

- KakaoTalk
- KakaoStory
- KakaoGroup
- Plain
- Brunch
- Path

Contents Platform

- Media Daum
- KakaoGame
- Emoticon Store
- KakaoPage
- KakaoMusic
- Daum tvPot
- Kakao TV
- 1 boon

Commerce Platform

- Gift Shop
- KakaoStyle

Marketing Platform

- Story Plus

Local Platform

- KakaoPlace

Advertise Platform

- Daum DDN

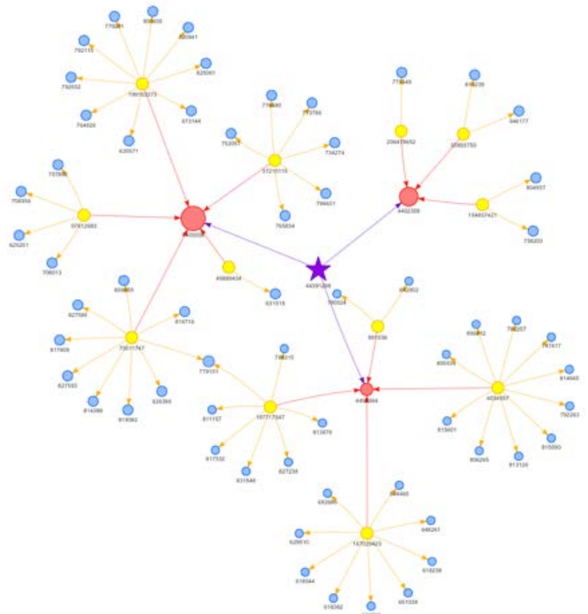
Statistics

kakao

1. total # of edges(activities + relations) >= 1 trillion
2. daily, new incoming edges in realtime >= 3 billion
3. daily, new edges that processed from batch process >= 50 billion
4. average query per minute >= 200 million. peak >= 400 million.
   under 50 ms.
5. 40% queries are 3 step query, 40% are 2 step, 20% are 1 step.

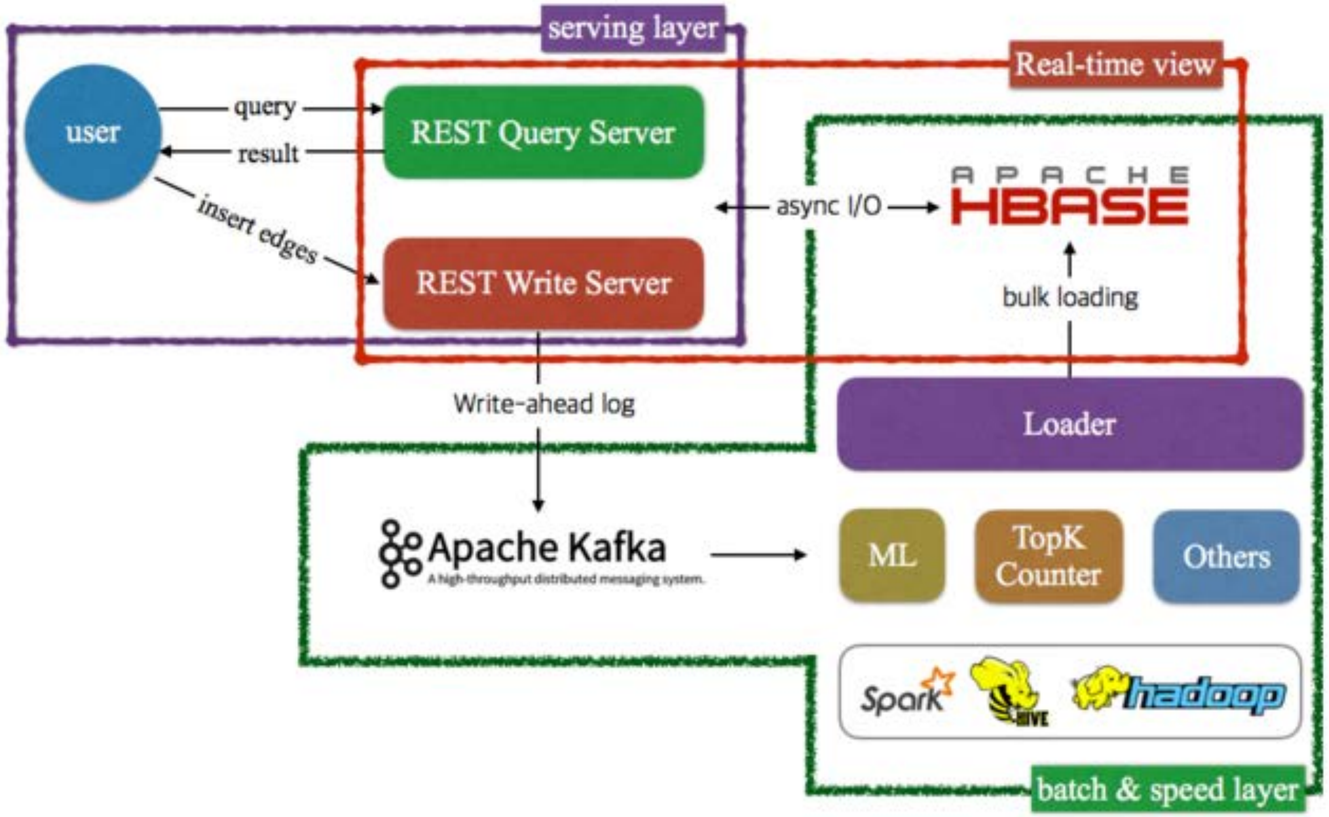
Operations

1. # of HBase region server = 40
2. # of query server = 70
3. # of write server = 20



Why S2Graph: Apache V2 License

kakao



Entire projects are Apache Incubator project.
This means S2Graph is open to anyone.