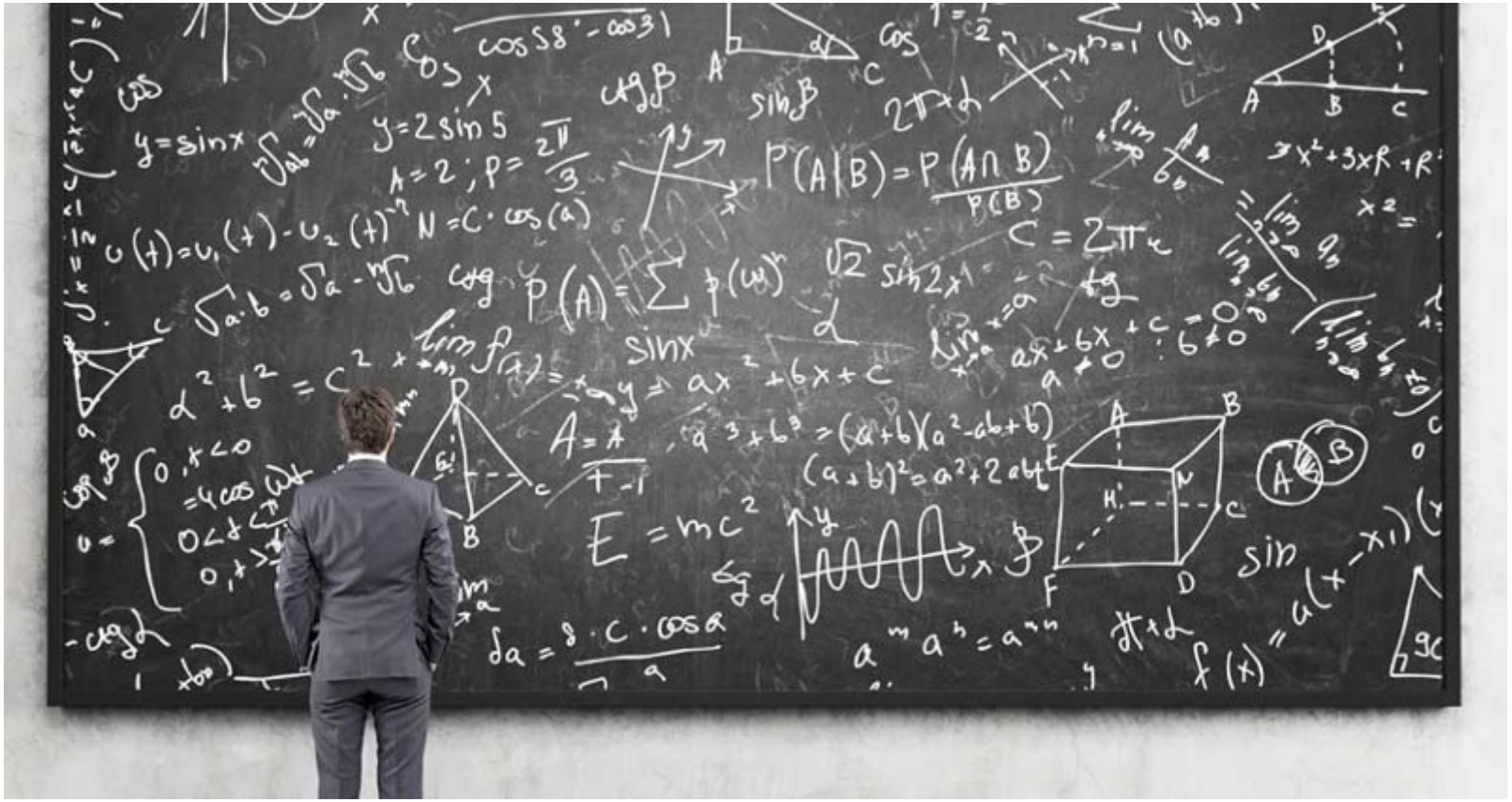
Sous le capot d’addok



Christian Quest - @cq94
christian.quest@data.gouv.fr
cquest@openstreetmap.fr

Histoire d’addok

- Besoin d’un géocodeur pour l’ouverture de la BAN (avril 2015)

- Plusieurs tentatives basées sur ElasticSearch
  - Résultats à 80 % « magiques »
  - Fonctionnement trop « boite noire »
  - Inadapté aux petits documents redondants

- Début du dev. par Yohan Boniface fin 2014...

Le cahier des charges...

- Rapide !
- Recherche d’adresses complète ou partielles (autocomplétion)
- Géocodage inverse
- Préférence par proximité
- Recherches avec « filtres »
- Fonctionnement compréhensible
- Rapide !

Recherche en texte intégral

- addok se base très peu sur la structure hiérarchique des adresses, il cherche des mots simplifiés (tokens)

  → *il peut chercher autre chose que des adresses ;)*

- Une recherche en 2 temps :
  - **Accumulation** successives de réponses potentielles
  - **Tri** des réponses pour sélectionner les meilleures

On remplit le seau et on sélectionne la crème...

La préparation...

- Nettoyage de la requête :
  - `Etalab, 20 av de Ségur, TSA 30719 75334 Paris Cedex 07`
  - `20 av de segur tsa 30719 75334 Paris Cedex 04`
  - `20 av de segur 75 paris`
- Traitement spécifique au pays et à la langue

  → Géré par plugins pour être adaptable et indépendant du moteur (addok-france + addok-fr)

Désabréviation et simplification

- Abréviations courantes et synonymes sont normalisés :

  20 av de segur 75 paris

  20 avenue de segur 75 paris

- Puis « tokenisé » :

  vin avenu de segur 75 paris

Nos 6 tokens initiaux à chercher dans l'index !

Remplissage du seau

- Tri des tokens par fréquence d’apparition
  - Peu courant : segur, 75, pari
  - Courant : avenue, de
  - Le N° potentiel : vin
  - Tokens inconnu : (aucun dans le cas présent)
- Recherches successives dans l’index redis
  - Sorted sets de Redis : ensembles d’éléments pondérés
  - ZUNIONSCORE : le secret d’addok ;)

Recherche de plus en plus « floue »

- Accumulation avec :
  - Recherche sur les tokens initiaux
  - Recherche avec les variations des tokens (« fuzzy »)

- Variations possibles :
  - Inversions de lettres, manques, substitutions :

    segur → sgeur, seur, sefur, segir

    pari → prai, pai, pati, paro

On accumule un maximum de 100 candidats

Les candidats

- Candidats sans numéro trouvés :

```
Villa de Ségur 75007 Paris
Avenue de Ségur 75015 Paris
Avenue de Ségur 75007 Paris
Rue Pérignon, Métro Segur 75015 Paris
Impasse des 3 soeurs 75011 Paris
Passage des 2 Soeurs 75009 Paris
Avenue de la Soeur Rosalie 75013 Paris
```

Il faut maintenant extraire la crème...

Comparaison avec la requête

- Le numéro est maintenant pris en compte
- Comparaison chaîne complète par :
  - Trigrammes (ne tient que peu compte de l’ordre)
  - Levenshtein (pour tenir compte de l’ordre)
- Trigrammes :
  - avenue de segur → ave, ven, nue, e d, de,…
  - Pourcentage de trigrammes communs

Le score final

- Préférence géographique :
  - Si demandée, un calcul de distance géographique par rapport au point initial est ajouté
- Importance :
  - Une valeur d’importance figurant dans le référentiel permet de trier les résultats en début d’autocomplétion (calculé en amont avant addok)
  - Av. des Champs Élysée ?

    → Paris avant Ponponne, Hirson ou Chadrac !

Temps de traitement...

- Nettoyage de la requête et tokens : 6 %
- Classement des tokens : 6 %
- Recherche et combinaison des tokens : 18 %
- Recherche « fuzzy » : 27 %
- Récupérations données (sqlite) : 16 %
- Calcul du score : 14 %

Temps moyen total constaté : 30ms

Le « shell » d’addok

DEMO !

Questions ?

- Article plus détaillé sur medium:
  - https://frama.link/addok-sous-le-capot

- Projet github :
  - https://github.com/addok

Christian Quest - @cq94
christian.quest@data.gouv.fr
cquest@openstreetmap.fr