cse.ust.hk

crawler

1. fetch with BFS (use HTML parser)

2. extract links to recursively process more pages.

3. build the file structure containing the parent/child link relation.

pages

crawler Service

indexer

defined in modals

1. remove all stop words from the file.
2. transform words into stems using Porter's algorithm (Perform stemming to the words)
3. Store the stems into inverted_titles and inverted_bodies
   ^ same type ^
4. support phrase search e.g. "hong kong"

inverted files

indexer service

uses

DAO Layer

Database Manager

mapdb

phase 1 tester program

↓

txt

app

provide APIs:

→ Method: GET
→ Path: api/search
→ Parameter: 1. query (user's search query)
  2. limit (# of returned result)
  3. offset (skip previous
→ Return: pageId, title, url, lastModified, snippet, score, pageSize, keywords, parentLinks, childLinks

→ Method: GET
→ Path: api/health
→ Parameter: None
→ Return: status (Boolean)

Search

Provide search service to app.
1. Retrieve inverted indices from DAO. → stem and remove stopword?
2. Compare list of query terms against the inverted indices.
3 rank according to a vector-space model $\alpha \approx 0.7$
   i.e. Final Score = α · Term Score + $(1-\alpha)$ · PageRank Score

Term-based:

$$\frac{tf \times idf}{\max(tf)}$$

Link-based:

$$PR(p_i) = (1-d) + \overset{0.85}{d} \cdot \sum_{p_j \in B(p_i)} \frac{PR(p_j)}{L(p_j)}$$

←

⚠ title matches boost the score

Database Schema

inverted_titles inverted_bodies

Word-ID → {Page-ID, Freq, pos}

← dummy

lightweight forward index

page ID → {keywords}

↳ top 10 stemmed keywords only.

Page properties

PageID → {title, URL, last-modified, size, content}

↳ we can check this before

one to one map

URL ⟷ pageID

Word ⟷ wordID

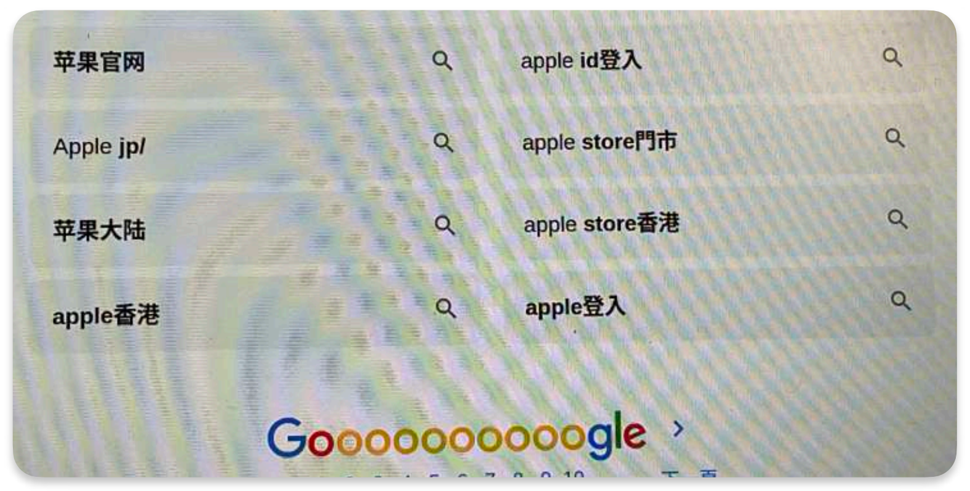
many-to-many

parent ⇄ child ↗ adjacency list representing the link structure

Bonus

F 1. relevance feedback

F 2. Allow user to:



苹果官网	apple id登入
Apple jp/	apple store門市
苹果大陆	apple store香港
apple香港	apple登入

Gooooooooooogle ›

F 3. UI-friendly ✓ (Theme: neuromorphism, colour: bg: white ↗ #f5f5f5, primary colour: Amethyst #9966cc, secondary colour: yellow #fff68f ↳ for highlighting, font: Poppins

F 4. Query history ✓ (up to 5 Query)

5. Other search engine func:
   1. word pos ✓ B
   2. Result sort by time ✓ F

B 6. Ranking (Page rank ✓ + Term based ✓)

B 7. do caching ✓, threading ✓ (speed)

FIFO, LFU, LRU?