**Гак Артем, МП КН – 2**

**Бази Знань**

Предметна область – теорія графів.

Дата сет – DBPedia

Вивантажено дані із теорії графів: 1058 елементів.

define input:ifp "IFP\_OFF" select ?s1 as ?c1, (bif:search\_excerpt (bif:vector ('GRAPH', 'THEORY'), ?o1)) as ?c2, ?sc, ?rank, ?g where {{{ select ?s1, (?sc \* 3e-1) as ?sc, ?o1, (sql:rnk\_scale (<LONG::IRI\_RANK> (?s1))) as ?rank, ?g where

{ quad map virtrdf:DefaultQuadMap

{ graph ?g

{ ?s1 ?s1textp ?o1 .

?o1 bif:contains '(GRAPH AND THEORY AND (CYCLIC OR ACYCLIC OR WEIGHTED OR UNWEIGHTED OR ORIENTED OR DIRECTED OR UNDIRECTED))' option (score ?sc) .

}

}

}

order by desc (?sc \* 3e-1 + sql:rnk\_scale (<LONG::IRI\_RANK> (?s1))) limit 2000 offset 0 }}}

[Permalink](https://dbpedia.org/sparql?default-graph-uri=&qtxt=%20%20%20%20%20define%20input%3Aifp%20%22IFP_OFF%22%20%20select%20%3Fs1%20as%20%3Fc1%2C%20(bif%3Asearch_excerpt%20(bif%3Avector%20('GRAPH'%2C%20'THEORY')%2C%20%3Fo1))%20as%20%3Fc2%2C%20%3Fsc%2C%20%3Frank%2C%20%3Fg%20where%20%7B%7B%7B%20select%20%3Fs1%2C%20(%3Fsc%20*%203e-1)%20as%20%3Fsc%2C%20%3Fo1%2C%20(sql%3Arnk_scale%20(%3CLONG%3A%3AIRI_RANK%3E%20(%3Fs1)))%20as%20%3Frank%2C%20%3Fg%20where%20%20%0A%20%20%7B%20%0A%20%20%20%20quad%20map%20virtrdf%3ADefaultQuadMap%20%0A%20%20%20%20%7B%20%0A%20%20%20%20%20%20graph%20%3Fg%20%0A%20%20%20%20%20%20%7B%20%0A%20%20%20%20%20%20%20%20%20%3Fs1%20%3Fs1textp%20%3Fo1%20.%0A%20%20%20%20%20%20%20%20%3Fo1%20bif%3Acontains%20%20'(GRAPH%20AND%20THEORY%20AND%20(CYCLIC%20OR%20ACYCLIC%20OR%20WEIGHTED%20OR%20UNWEIGHTED%20OR%20ORIENTED%20OR%20DIRECTED%20OR%20UNDIRECTED))'%20%20option%20(score%20%3Fsc)%20%20.%0A%20%20%20%20%20%20%20%20%0A%20%20%20%20%20%20%7D%0A%20%20%20%20%20%7D%0A%20%20%20%20%0A%20%20%7D%0A%20order%20by%20desc%20(%3Fsc%20*%203e-1%20%2B%20sql%3Arnk_scale%20(%3CLONG%3A%3AIRI_RANK%3E%20(%3Fs1)))%20%20limit%202000%20%20offset%200%20%7D%7D%7D%20%0A&format=application%2Fsparql-results%2Bjson&timeout=30000&signal_void=on)

Приклад елемента:

"name": "(a,\_b)-decomposition",

      "uri": "http://dbpedia.org/resource/(a,\_b)-decomposition",

      "description": "in graph theory the a b decomposition of an undirected graph is a partition of its edges into a 1 sets each nducing a forest except one which induces a graph with maximum degree if this graph is also a forest then we call this a f a b",

Обробка дата сету – python script:

Опис розбито на токени (key\_word) і далі побудовано інвертований індекс: key\_word -> Frequency (кількість у всій колекції)

Кількість унікальних key\_words: 2858

{

      "name": "(a,\_b)-decomposition",

      "uri": "http://dbpedia.org/resource/(a,\_b)-decomposition",

      "description": "in graph theory the a b decomposition of an undirected graph is a partition of its edges into a 1 sets each nducing a forest except one which induces a graph with maximum degree if this graph is also a forest then we call this a f a b",

      "key\_words": {

        "graph": 1989,

        "theory": 1969,

        "decomposition": 27,

        "undirected": 292,

        "partition": 35,

        "edges": 204,

        "sets": 36,

        "nducing": 2,

        "forest": 15,

        "except": 3,

        "induces": 2,

        "maximum": 49,

        "degree": 68,

        "also": 114,

        "call": 2

      }

    },

І потім додано параметр freq\_percent – відсоток входження цього слова:

Сума всіх frequencies: 22816

102/22816 = 0.00447054698457223

Key\_word = problem:(frequency: 102, freq\_percent: 0.00447054698457223)

Завантаження в neo4j – load script, plugin from json

CALL apoc.load.json("file:///converted\_data.json")

YIELD value

MERGE (c:GraphConcept {name: value.name})

SET c.uri = value.uri, c.description = value.description

WITH c, value

UNWIND keys(value.key\_words) AS keyword

MERGE (kw:KeyWord {name: keyword})

ON CREATE SET kw.frequency = value.key\_words[keyword]

MERGE (c)-[:Has\_KeyWord]->(kw);

A screenshot of a computer

Description automatically generated

Використання, запити

1. **Знайти ключові слова до заданого концепту.**

Наприклад: користувач пише статтю і хоче підібрати ключові слова

MATCH (gc:GraphConcept {name: "Apex\_graph"})-[:Has\_KeyWord]-(kw:KeyWord)

RETURN gc, kw;

A screenshot of a computer

Description automatically generated

1. **Знайти концепти за ключовими словами**

Наприклад: користувач цікавиться які Концепти мають задані «властивості»

MATCH (kw:KeyWord)<-[:Has\_KeyWord]-(gc:GraphConcept)

WHERE kw.name IN ["clique","complexity","coloring"]

RETURN kw, gc;

Наприклад, користувач хоча подосліджувати речі пов'язані із кліками, складністю і розфарбуванням.

Результати:

«FNP\_(complexity)» = ["clique","complexity","coloring"]

«Clique-width», «Tardos\_function» = ["clique","complexity"]

«Interval\_graph»= ["clique"," coloring "]

«Daniel\_Kráľ»= ["complexity "," coloring "]

A screenshot of a computer screen

Description automatically generated

1. **Найбільш unique пов’язані графи**

Знайти пари графів, які мають спільні key-words із найменшою частотою на всю базу знань.

Тобто такі-собі пари унікально пов’язаних графів.

MATCH (gc1:GraphConcept)-[:Has\_KeyWord]->(kw:KeyWord)<-[:Has\_KeyWord]-(gc2:GraphConcept)

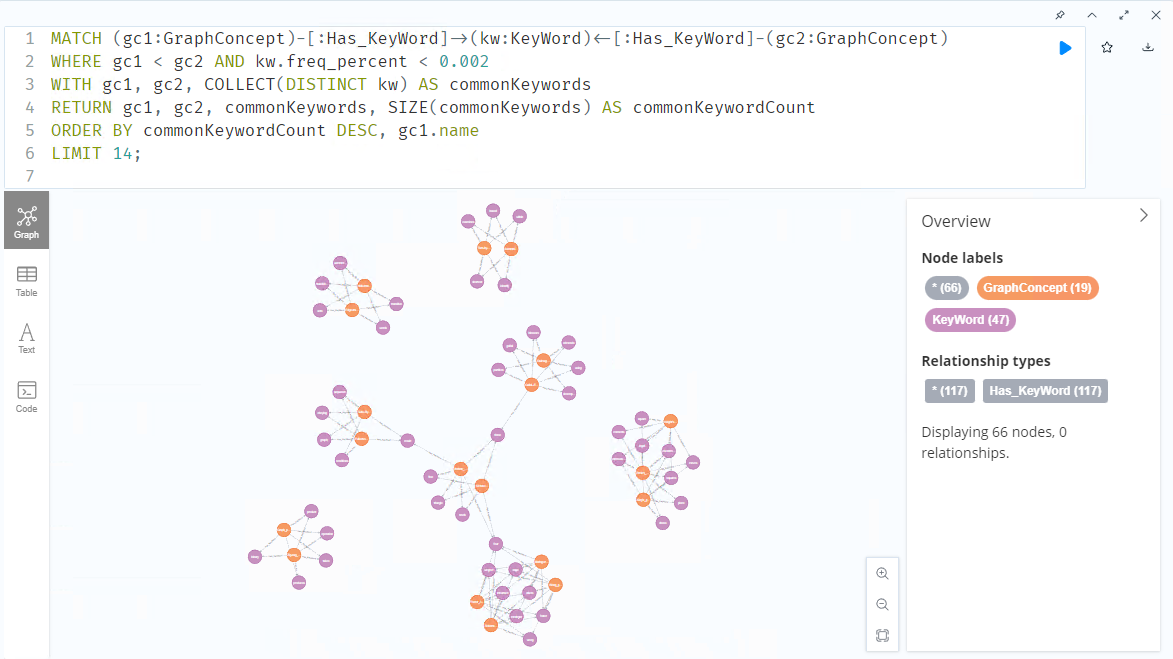
WHERE gc1 < gc2 AND kw.freq\_percent < 0.002

WITH gc1, gc2, COLLECT(DISTINCT kw) AS commonKeywords

RETURN gc1, gc2, commonKeywords, SIZE(commonKeywords) AS commonKeywordCount

ORDER BY commonKeywordCount DESC, gc1.name

LIMIT 14;



|  |  |  |  |
| --- | --- | --- | --- |
| gc1.name | gc2.name | commonKeywords | Count |
| "Foster\_cage" | "Robertson–Wegner\_graph" | ["cage", "wong", "four", "robertson", "meringer", "wegner", "foster", "others"] | 8 |
| "Foster\_cage" | "Meringer\_graph" | ["cage", "four", "robertson", "meringer", "wegner", "foster", "others"] | 7 |
| "Foster\_cage" | "Wong\_graph" | ["cage", "wong", "four", "meringer", "wegner", "foster", "others"] | 7 |
| "King's\_graph" | "Rook's\_graph" | ["represents", "squares", "chess", "chessboard", "piece", "moves", "legal"] | 7 |
| "Knight's\_graph" | "Rook's\_graph" | ["square", "represents", "connects", "squares", "chessboard", "moves", "legal"] | 7 |
| "Meringer\_graph" | "Robertson–Wegner\_graph" | ["cage", "four", "robertson", "meringer", "wegner", "foster", "others"] | 7 |
| "Robertson–Wegner\_graph" | "Wong\_graph" | ["cage", "wong", "four", "meringer", "wegner", "foster", "others"] | 7 |
| "Dulmage–Mendelsohn\_decomposition" | "Gallai–Edmonds\_decomposition" | ["decomposition", "partition", "using", "blossom", "gallai", "edmonds"] | 6 |
| "Meringer\_graph" | "Wong\_graph" | ["cage", "four", "meringer", "wegner", "foster", "others"] | 6 |
| "Automotive\_navigation\_  system" | "Turn-by-turn\_navigation" | ["within", "based", "identify", "examines", "shortest"] | 5 |
| "Edge-transitive\_graph" | "Half-transitive\_graph" | ["automorphism", "transitive", "words", "transitively", "acts"] | 5 |
| "Fulkerson–Chen–Anstee\_theorem" | "Gale–Ryser\_theorem" | ["graph;", "result", "sequence", "obeying", "conditions"] | 5 |
| "Graph\_product" | "Zig-zag\_product" | ["product", "produces", "binary", "operation", "takes"] | 5 |
| "Grötzsch's\_theorem" | "Herbert\_Grötzsch" | ["three", "free", "triangle", "four", "tzsch"] | 5 |

1. **Концепти які мають мало ключових слів (найменш вивчені)**

MATCH (n)-[:Has\_KeyWord]->(w)

RETURN n, COLLECT(w), COUNT(w) as s

ORDER BY s DESC

LIMIT \_

|  |  |
| --- | --- |
| n.name |  |
| "Jinyoung\_Park\_(mathematician)" | 2 ["graph", "theory"] |
| "Witold\_Lipski" | 2 ["graph", "theory"] |
| "Italo\_Jose\_Dejter" | 2 ["graph", "theory"] |
| "Richard\_A.\_Brualdi" | 2 ["graph", "theory"] |
| "Sajal\_K.\_Das" | 2 ["graph", "theory"] |
| "End\_(graph\_theory)" | 2 ["graph", "theory"] |
| "Henda\_Swart" | 2 ["graph", "theory"] |
| "Ray\_(graph\_theory)" | 2 ["graph", "theory"] |
| "Renu\_C.\_Laskar" | 2 ["graph", "theory"] |
| "Arc\_(graph\_theory)" | 2 ["graph", "theory"] |
| "Saidur\_Rahman\_(professor)" | 2 ["graph", "theory"] |
| "Category:Graph\_minor\_theory" | 3 ["minor", "graph", "theory"] |

1. **Слова які належать малій кількості концептів (найменш вивчені)**

MATCH (n)-[:Has\_KeyWord]->(w)

RETURN n, COLLECT(w), COUNT(w) as s

ORDER BY s DESC

LIMIT 7

|  |  |  |
| --- | --- | --- |
| w.name | s | COLLECT(n.name) |
| "hexagonal" | 1 | ["26-fullerene\_graph"] |
| "taxicab" | 1 | ["Absolute\_difference"] |
| "fullerene" | 1 | ["26-fullerene\_graph"] |
| "induces" | 1 | ["(a,\_b)-decomposition"] |
| "call" | 1 | ["(a,\_b)-decomposition"] |
| "pentagonal" | 1 | ["26-fullerene\_graph"] |
| "negations" | 1 | ["2-satisfiability"] |

1. **Знайти GraphConcepts, що мають найбільше KeyWords**

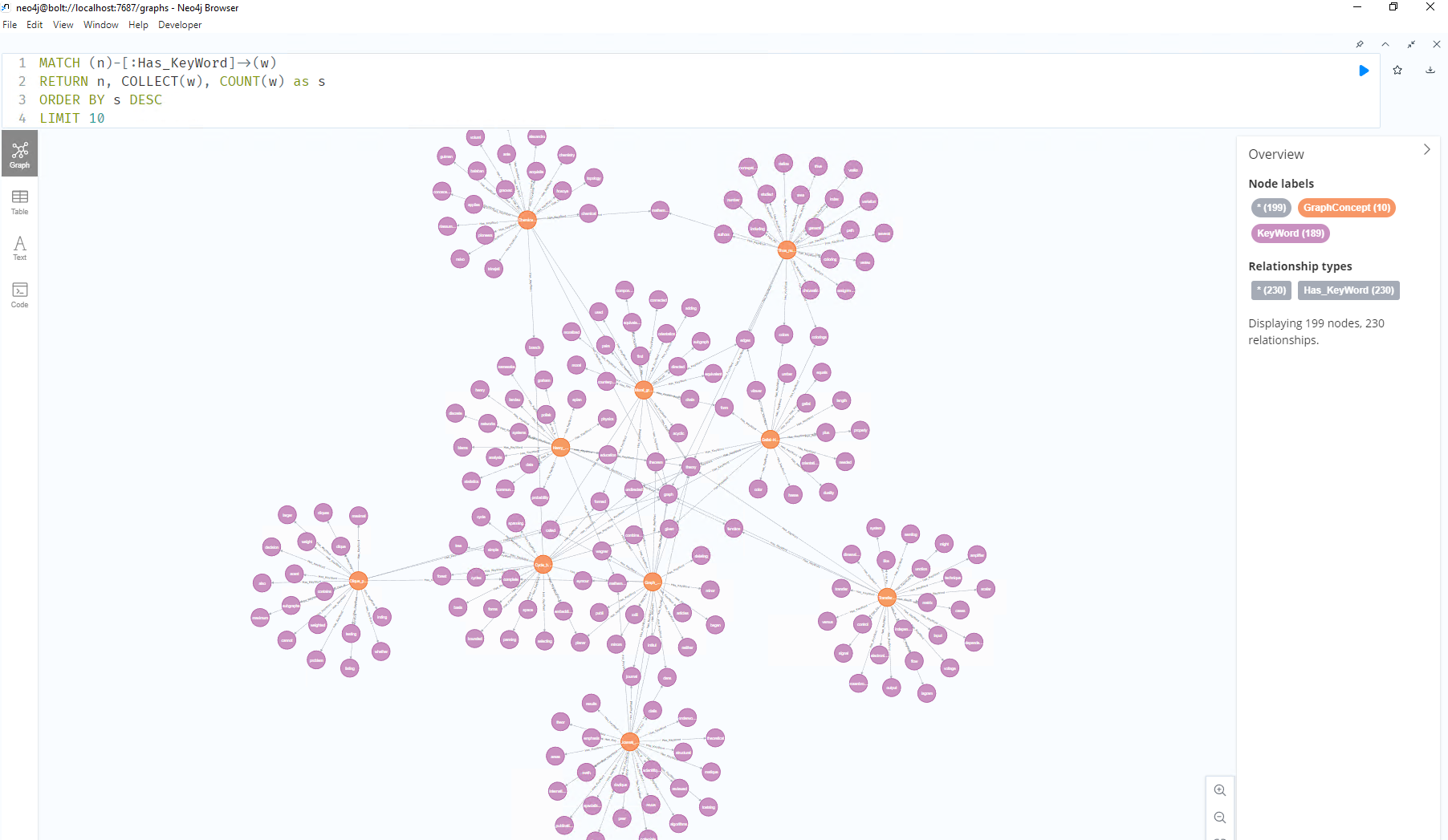
MATCH (n)-[:Has\_KeyWord]->(w)

RETURN n, COLLECT(w), COUNT(w) as s

ORDER BY s DESC

LIMIT 10

|  |  |
| --- | --- |
| Concept name | Count related keywords |
| "Journal\_of\_Graph\_Theory" | 27 |
| "Transfer\_function" | 27 |
| "Thue\_number" | 24 |
| "Moral\_graph" | 23 |
| "Graph\_minor" | 22 |
| "Chemical\_graph\_theory" | 22 |
| "Henry\_O.\_Pollak" | 22 |
| "Cycle\_basis" | 21 |
| "Clique\_problem" | 21 |
| "Gallai–Hasse–Roy–Vitaver\_theorem" | 21 |



A screenshot of a computer

Description automatically generated

1. **Знайти KeyWords, що мають найбільше GraphConcepts**

MATCH (n)-[:Has\_KeyWord]->(w)

RETURN w.name, COUNT(n) as s, COLLECT(n.name)

ORDER BY s DESC

LIMIT 10

|  |  |
| --- | --- |
| Key\_word | Count related concepts |
| "graph" | 1049 |
| "theory" | 1040 |
| "mathematical" | 199 |
| "vertices" | 193 |
| "undirected" | 149 |
| "field" | 134 |
| "vertex" | 130 |
| "edges" | 113 |
| "mathematics" | 111 |
| "number" | 104 |

A screenshot of a computer

Description automatically generated