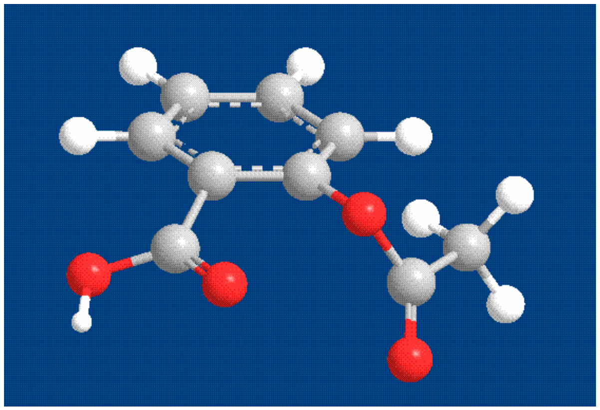
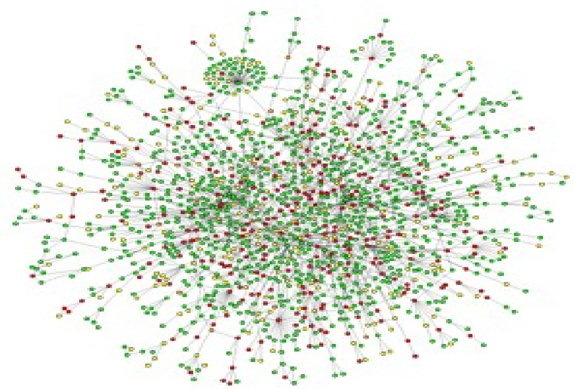
SEARCHING SUBSTRUCTURES WITH SUPERIMPOSED DISTANCE

Xifeng Yan, Feida Zhu
Jiawei Han, Philip S. Yu
University of Illinois at Urbana-Champaign
IBM T. J Watson Research Center

GRAPHS ARE EVERYWHERE

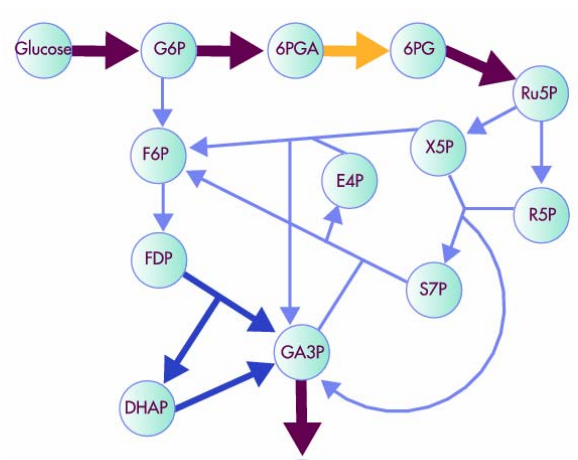


Aspirin

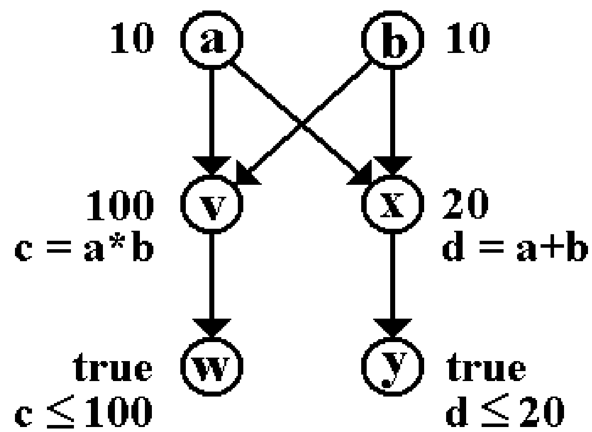


from H. Jeong et al Nature 411, 41 (2001)

Yeast Protein Interaction Network



Metabolic Network



Dependency Graph

GRAPH DATA

- Chem-informatics: chemical compounds
- Bioinformatics: protein structures, protein interaction networks, biological pathways, metabolic networks, …
- Computer Vision: object models
- Software: program dependency graph, flow graph,…
- Social network
- Workflow

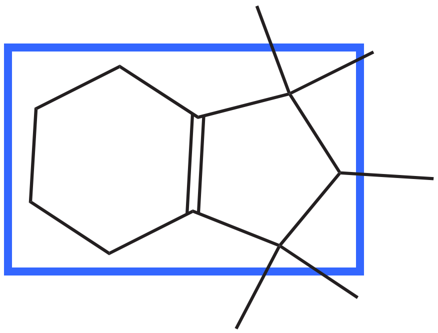
GRAPH INFORMATION SYSTEM

Applications	
• Characterize graph objects • Build indices for graph search • Extract biologically conserved modules • Discriminate drug complexes • Classify protein structures • Cluster gene networks • Detect anomaly in program flows • Graph registration system	

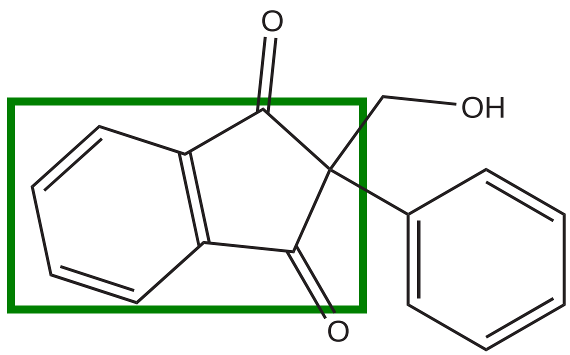
Graph Mining	Graph Search
finding hidden patterns	processing graph queries

GRAPH SEARCH

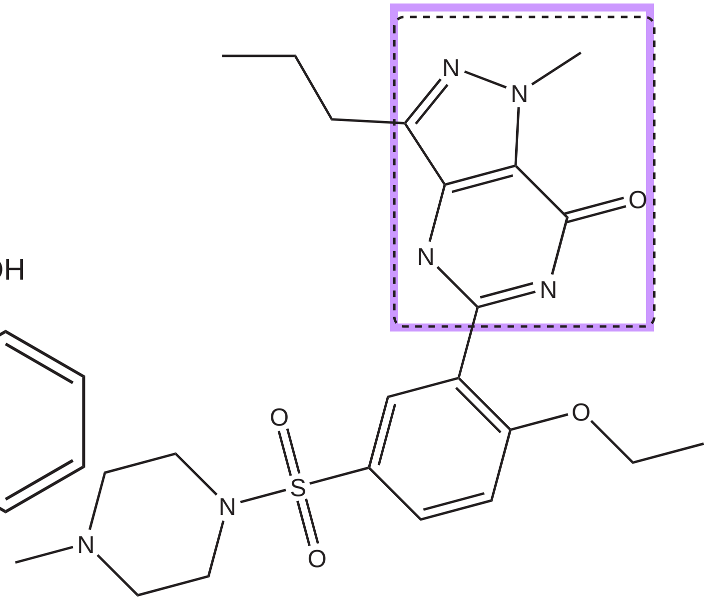
- **Chemical Compounds**



(a) 1H-Indene

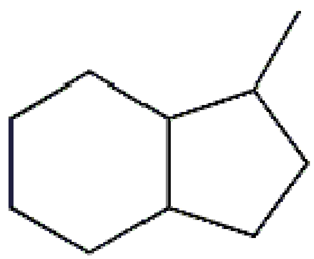


(b) Omephine



(c) Digitoxigenin

- **Query Graph**

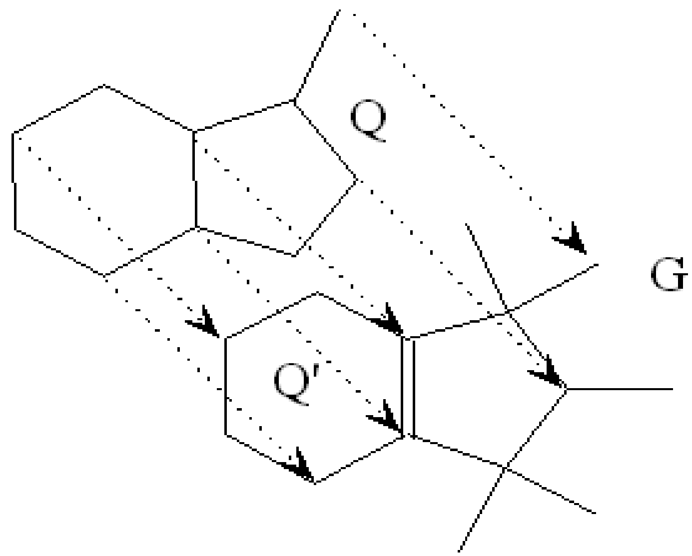


VARIETY OF GRAPH SEARCH

- Full structure search
- Substructure search [Shasha et al. PODS'02, Yan et al. SIGMOD'04]
- Approximate substructure search [Yan et al. SIGMOD'05]
- Substructure search with constraints
  - Superimposed distance [this work, ICDE'06]
  - Other varieties

SUPERIMPOSED DISTANCE

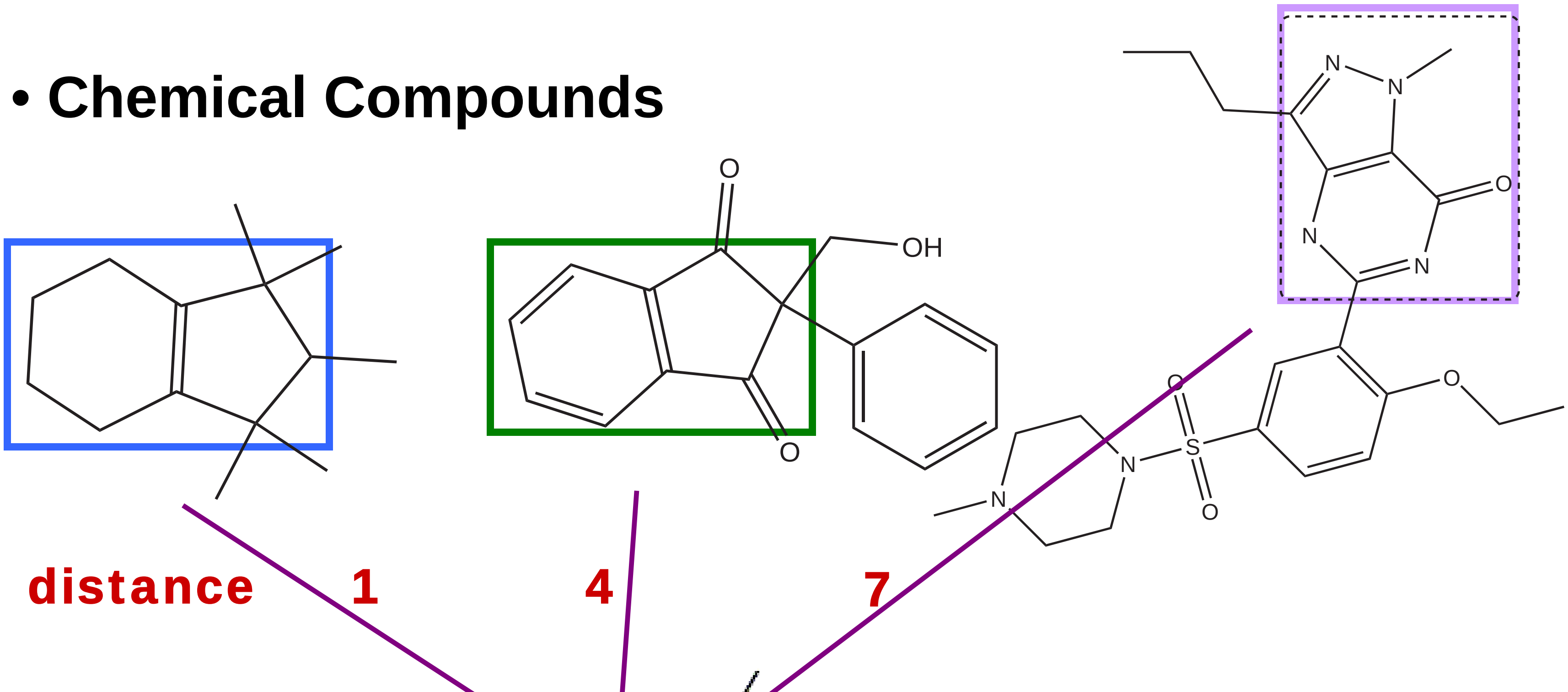
Same Topological Structure
But different Labels



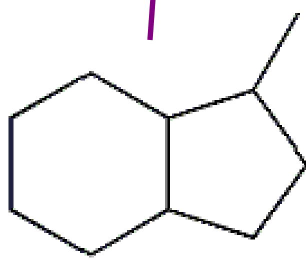
$$\mathrm{MD} = \sum_{v'=f(v)} \mathbf{D}(l(v), l'(v')) + \sum_{e'=f(e)} \mathbf{D}(l(e), l'(e'))$$

SUPERIMPOSED DISTANCE

- Chemical Compounds



- Query Graph



MINIMUM SUPERIMPOSED DISTANCE

Given two graphs, Q and G, let M be the set of subgraphs in G that are isomorphic to Q. The minimum superimposed distance between Q and G is the minimum distance between Q and Q' in M.

$$d(Q,G) = \min_{Q' \in M} d(Q,Q'),$$

where $d(Q,Q')$ is a distance function of two isomorphic graphs Q and Q'.

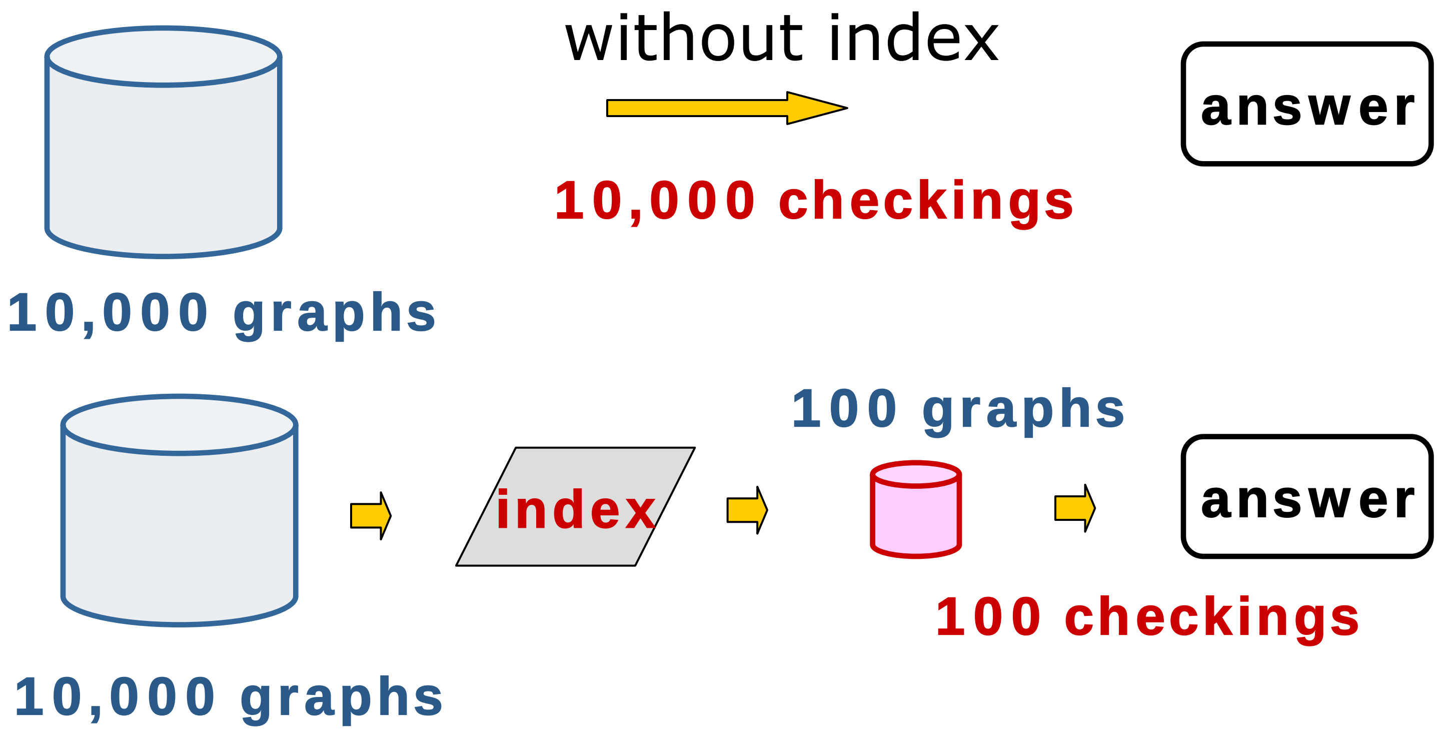
SUBSTRUCTURE SEARCH WITH SUPERIMPOSED DISTANCE (SSSD)

Given a set of graphs $D=\{G_1, G_2, \ldots, G_n\}$ and a query graph Q, SSSD is to find all G_i in D such that

$$d(Q, G_i) \leq \sigma$$

INDEXING GRAPHS

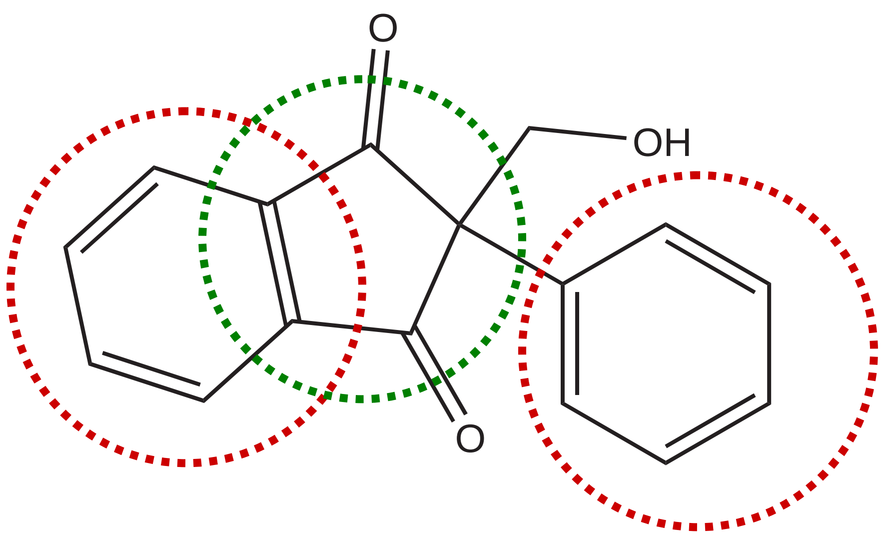
- Indexing is crucial



FEATURE-BASED INDEX

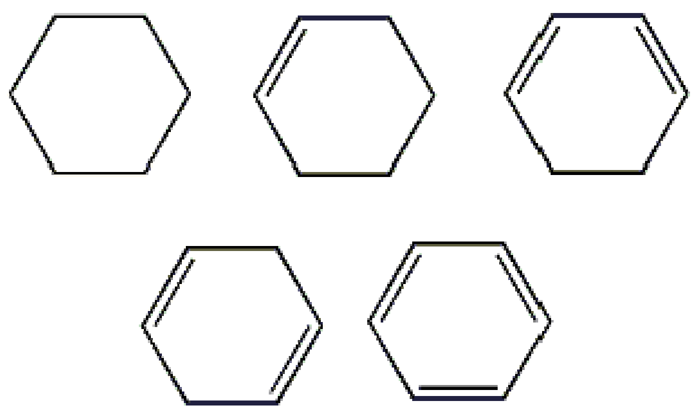
Feature:

1. Paths (Shasha et al. PODS'02)
2. Discriminative Frequent Substructures (Yan et al. SIGMOD'04)

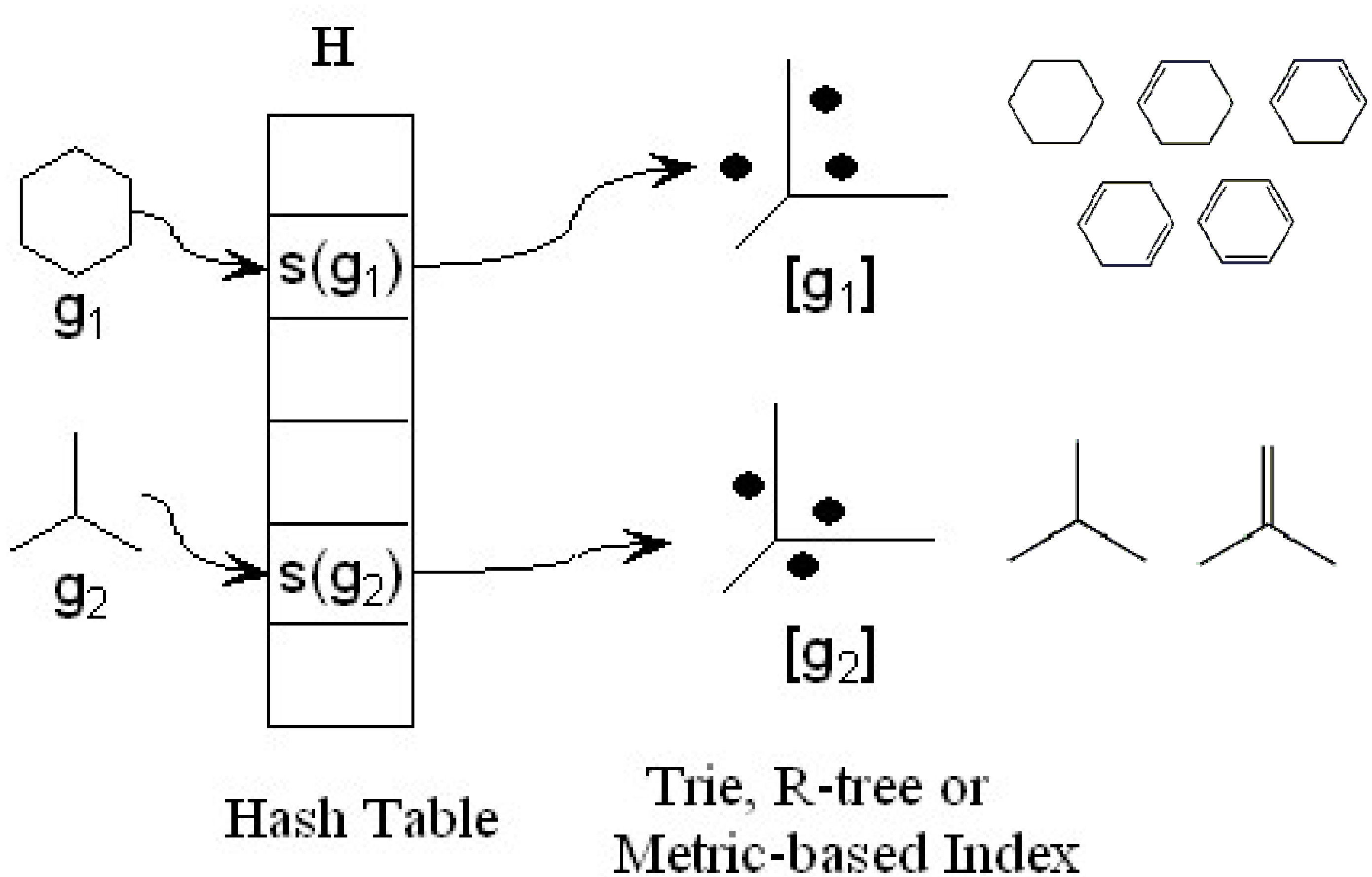


STRUCTURAL EQUIVALENCE CLASS

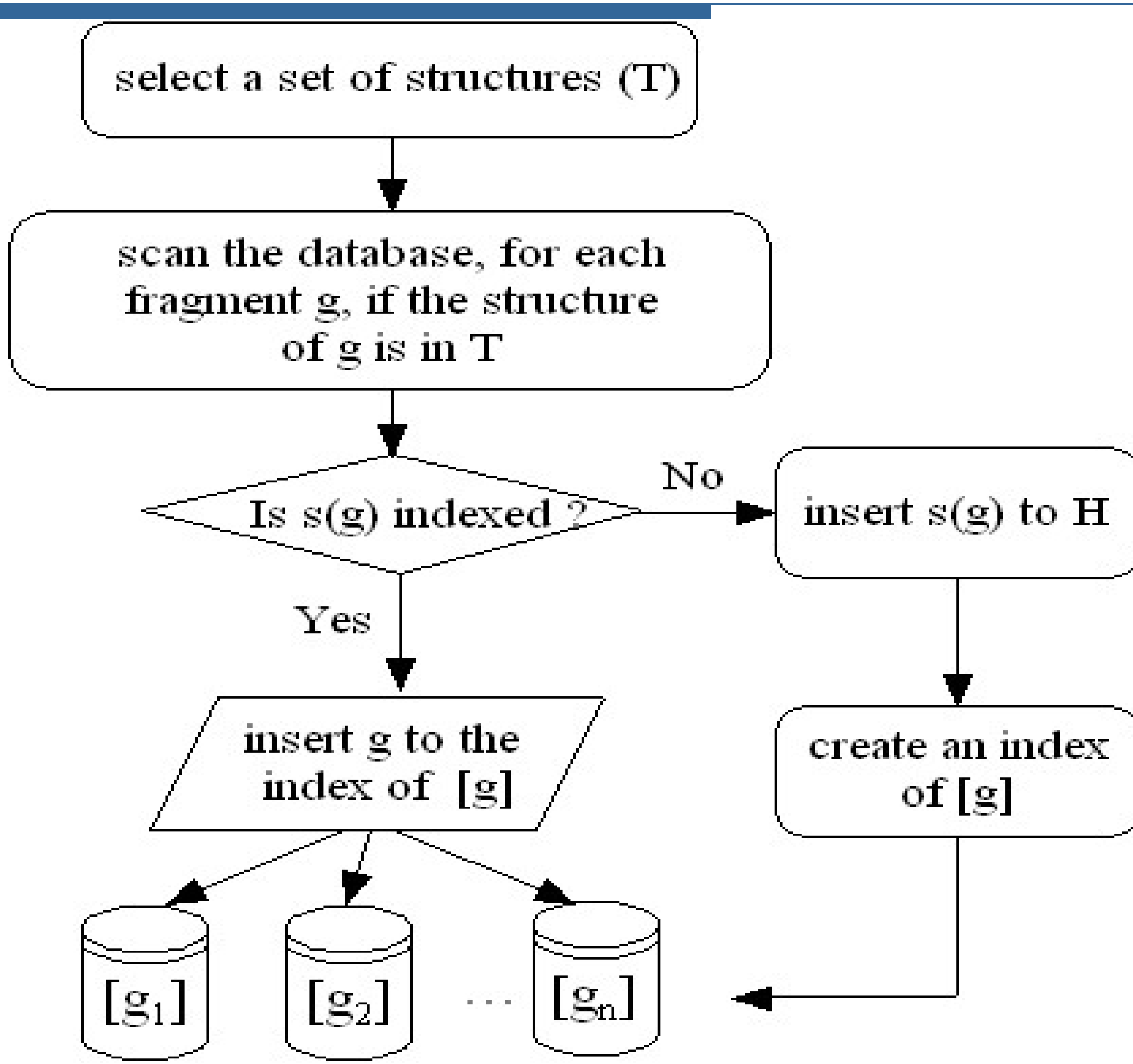
- Graphs G and G′ belong to the same equivalence class if and only if G is isomorphic to G′. The structural equivalence class of G is written [G]



THE INDEX STRUCTURE



INDEX CONSTRUCTION



(c) X. Yan 2006

PARTITION-BASED SEARCH

- We partition a query graph Q into **non-overlapping** indexed features f_1, f_2, ..., f_m, and use them to do pruning. If the distance function satisfies the following inequality,

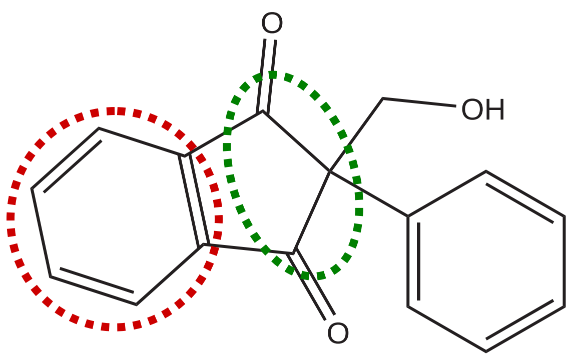
$$\sum_{i=1}^{m} d(f_i, G) \le d(Q, G)$$

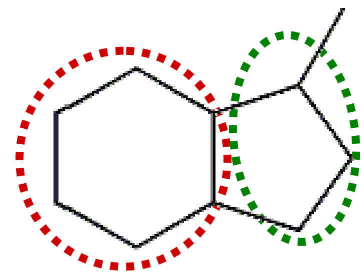
we can get the lower bound of the superimposed distance between Q and G by adding up the superimposed distance between f_i and G.

MULTIPLE PARTITIONS

Target graph G

Query graph Q

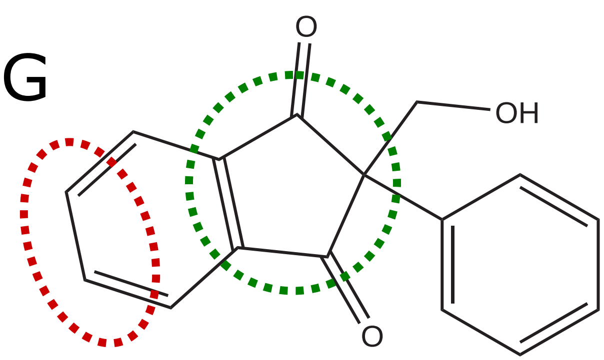




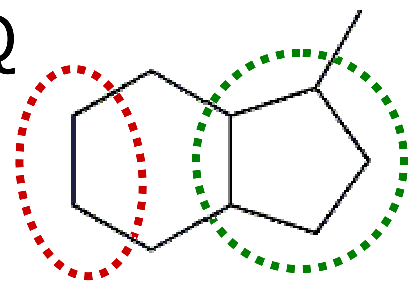
Partition I

Hexagon + Path

G



Q

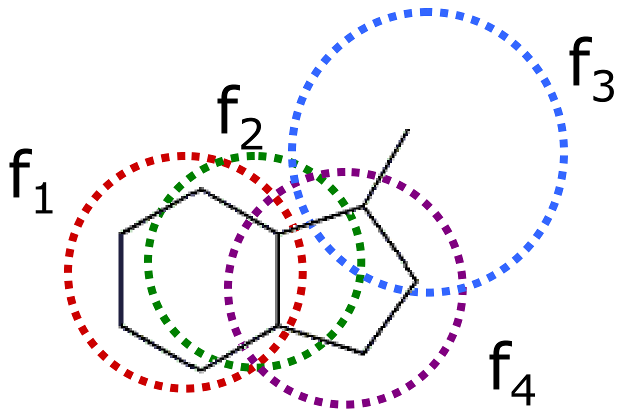


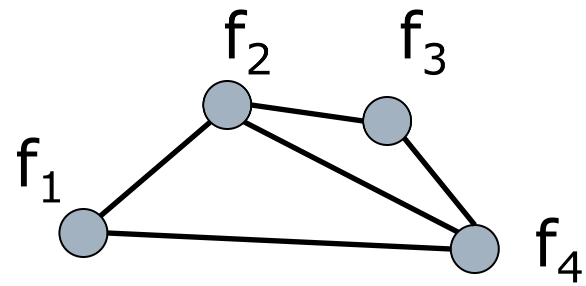
Partition II

Pentagon + Path

OVERLAPPING RELATION GRAPH

Query graph Q





node: feature

edge: overlapping

node weight: minimum distance between f_i and G, $d(f_i, G)$

SEARCH OPTIMIZATION

Given a graph Q=(V, E), a partition of G is a set of subgraphs $\{f_1, f_2, \ldots, f_m\}$ such that

$$V(f_i) \subseteq V \ and \ V(f_i) \cap V(f_j) = \emptyset$$

for any i!= j.

GIVEN A GRAPH G, OPTIMIZE

$$P_{opt(Q,G)} = \arg\max_{P} \sum_{i=1}^{m} d(f_i, G)$$

FROM ONE TO MULTIPLE

GIVEN A GRAPH G, OPTIMIZE

$$P_{opt(Q,G)} = \arg \max_P \sum_{i=1}^{m} d(f_i, G)$$

For one graph G, select one partition

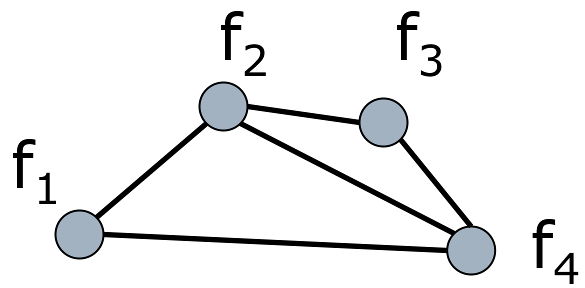
For another graph G', select another partition?

GIVEN A SET OF GRAPHS , OPTIMIZE

$$P_{opt(Q,G)} = \arg \max_P \sum_{j=1}^{n} \sum_{i=1}^{m} d(f_i, G_j)$$

$$= \arg \max_P \sum_{i=1}^{m} \boxed{\sum_{j=1}^{n} d(f_i, G_j)}$$

ACROSS MULTIPLE GRAPHS

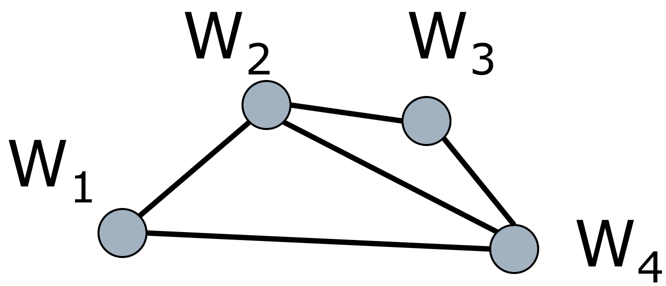


node weight is redefined

Using average minimum distance between a feature f and the graphs G_i in the database, written as

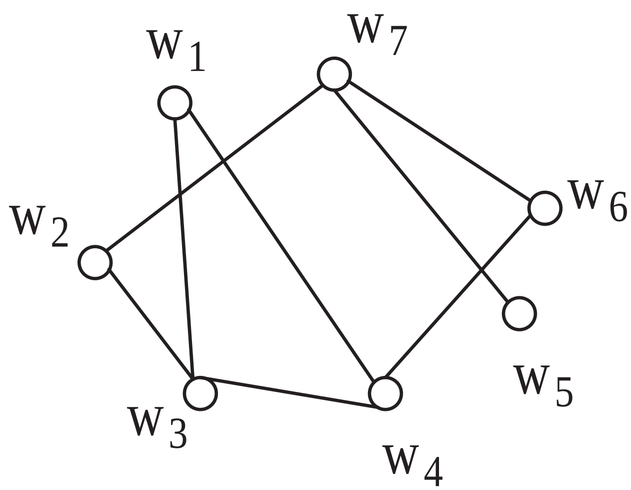
$$w(f) = \frac{\sum_{i=1}^{n} d(f, G_i)}{n}$$

MAXIMUM WEIGHTED INDEPENDENT SET

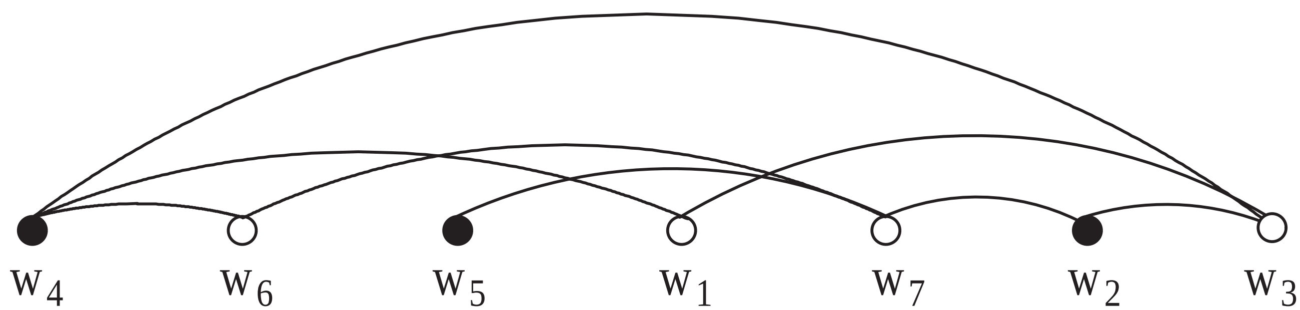


[THEOREM]
Index-based Partition Optimization is NP-hard.

GREEDY SOLUTION



w4 ≥ w6 ≥ w5 ≥ w1 ≥ w7 ≥ w2 ≥ w3



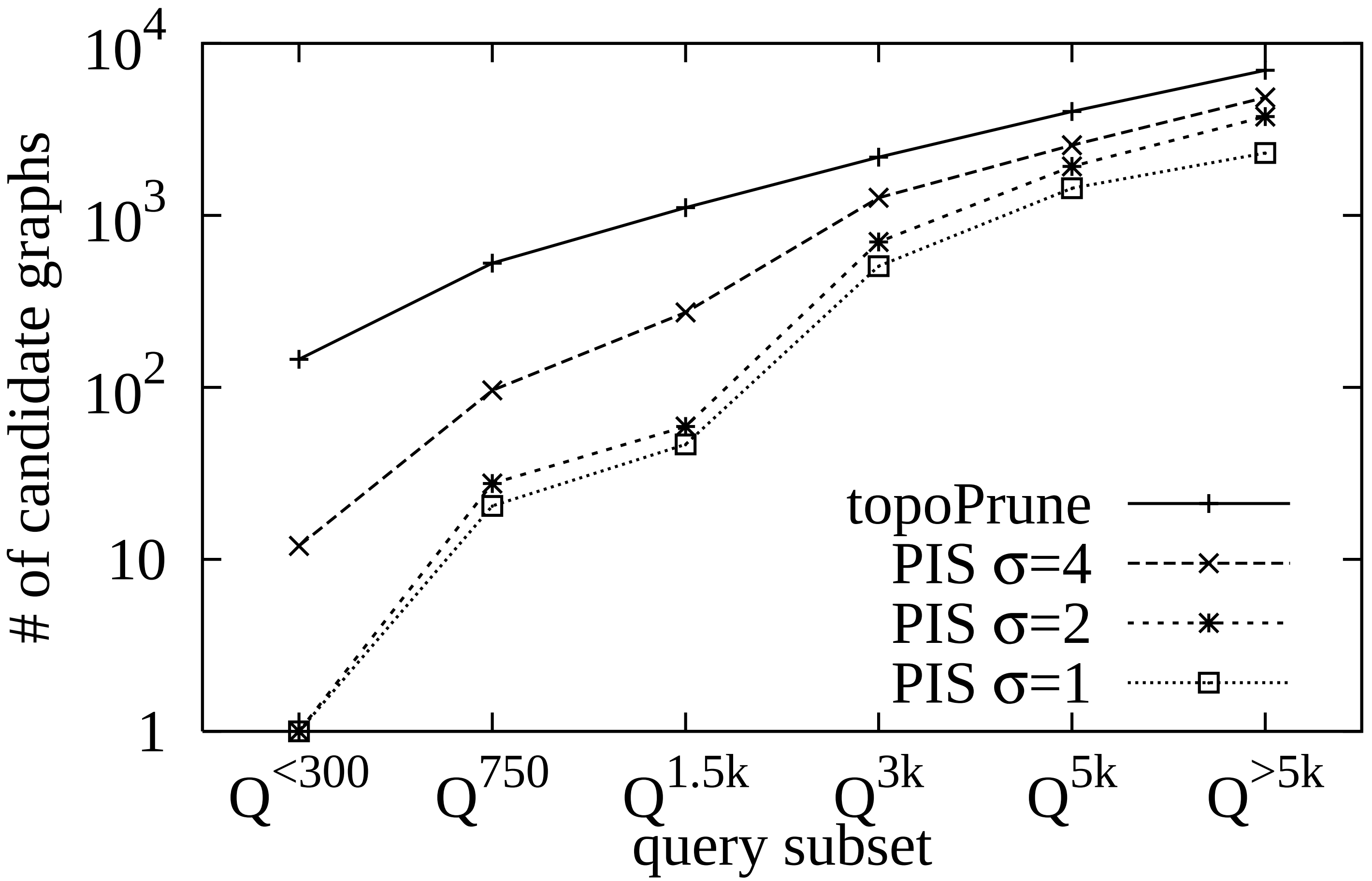
Experiment Dataset

- The real dataset is from an AIDS antiviral screen database containing the structures of chemical compounds.
- This dataset is available on the website of the Developmental Therapeutics Program (NCI/NIH).
- In this dataset, thousands of compounds have been checked for evidence of anti-HIV activity. The dataset has around 44,000 structures.

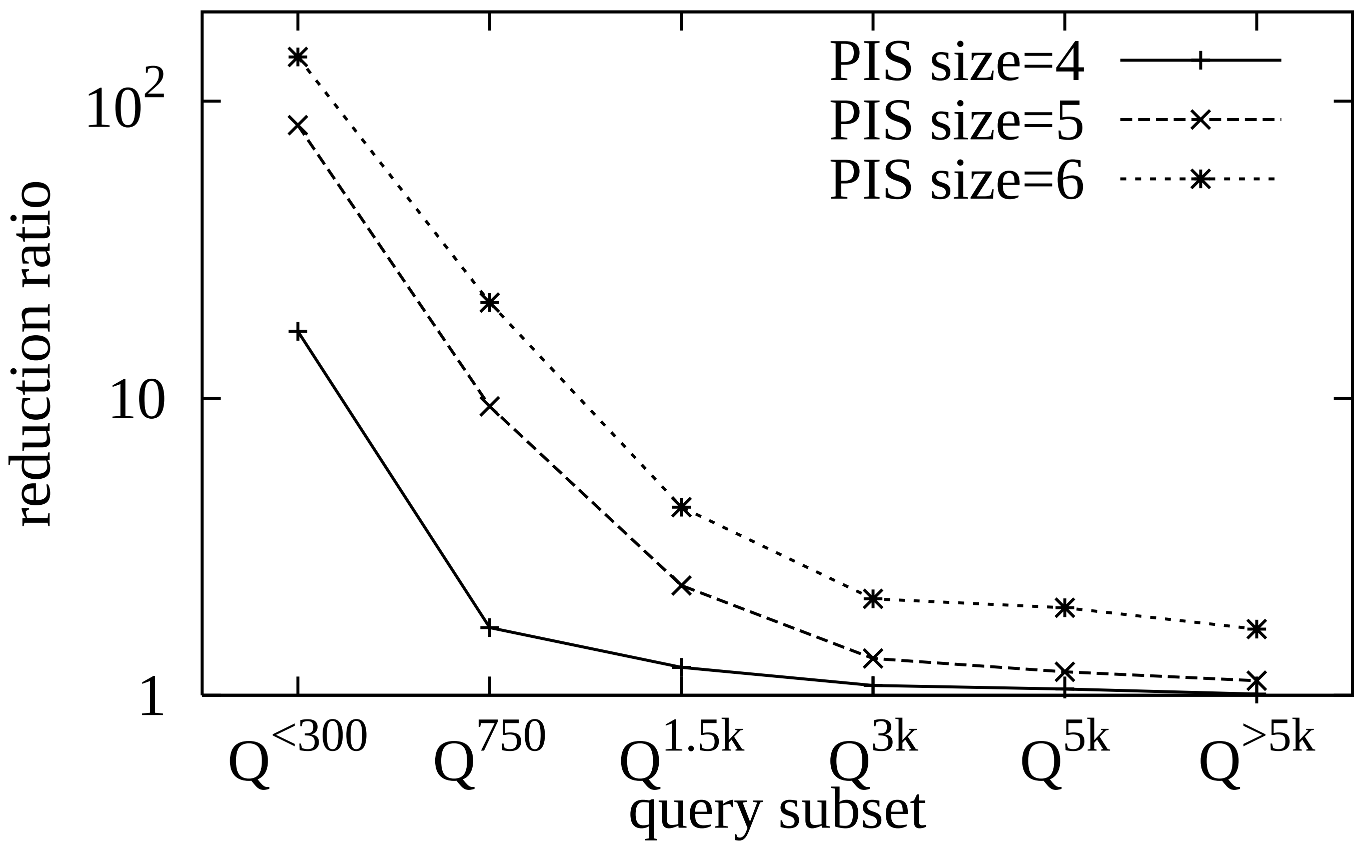
Experiment Setting

- We build topoPrune and PIS based on the gIndex (SIGMOD'04). gIndex first mines frequent structures and then retains discriminative ones as indexing features.
- topoPrune and PIS are implemented in C++ with standard template library.
- All of the experiments are done on a 2.5GHZ, 1GB memory, Intel Xeon PC running Fedora 2.0.

Pruning Efficiency



Efficiency vs. Fragment Size



CONCLUSIONS

- A substructure search problem with additional similarity requirements
- A problem as a component in our graph information system
- Approach: feature-based index and partition-based search
- HIGHLIGHT: select "discriminative" features in a query space for search efficiency

Thank You