Applications of Matrix Algebra

Vector Space Model for text data, Latent Semantic Indexing (LSI) for text searches, matrix representation for graph/network data, Recommender Systems as matrix factorization

Nagiza F. Samatova, samatova@csc.ncsu.edu
Professor, Department of Computer Science
North Carolina State University

NC STATE Executive Education

NC STATE UNIVERSITY Department of Computer Science

Text Analytics

VECTOR SPACE MODEL

Text Analytics Example

Titles
C1: *Human* machine *interface* for LAB ABC *computer* applications
C2: A *survey* of *user* opinion of *computer system response time*
C3: The *EPS user interface* management system
C4: *System* and *human system* engineering testing of *EPS*
C5: Relation of *user*-percieved *response time* to error measurement
M1: The generation of random, binary, unordered *trees*
M2: the intersection *graph* of paths in *trees*
M3: *Graph minors* IV: Widths of *trees* and well-quasi-ordering
M4: *Graph minors*: A *survey*

Italicized words occur and multiple docs and are indexed

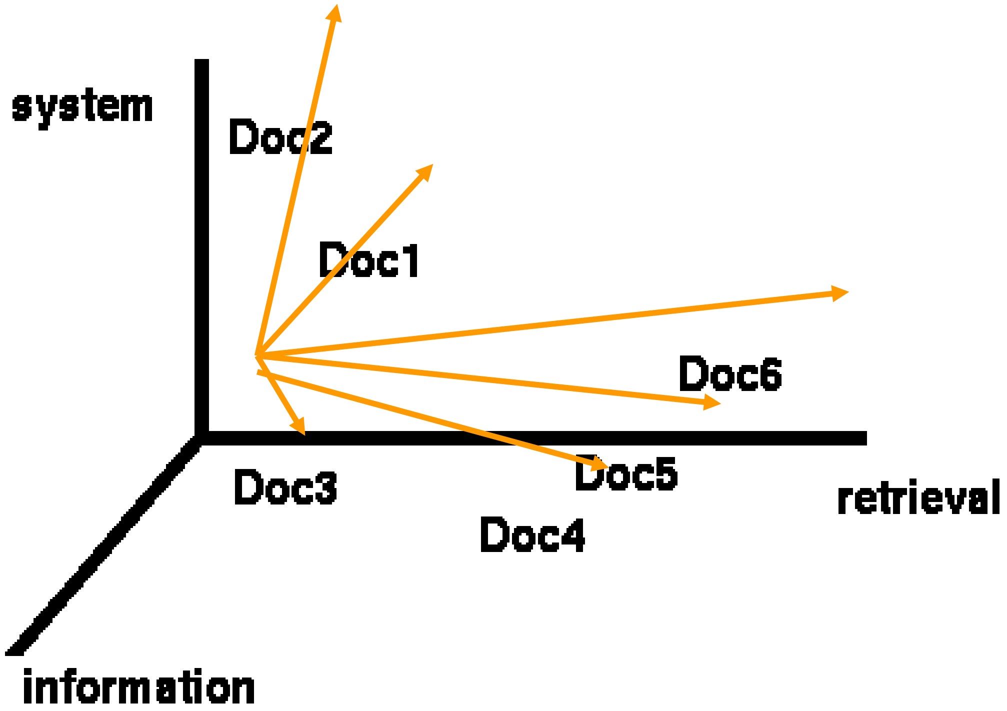
Term-Document Matrix

Terms	Documents								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
Human	1	0	0	1	0	0	0	0	0
Interface	1	0	1	0	0	0	0	0	0
Computer	1	1	0	0	0	0	0	0	0
User	0	1	1	0	1	0	0	0	0
System	0	1	1	2	0	0	0	0	0
Response	0	1	0	0	1	0	0	0	0
Time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
Survey	0	1	0	0	0	0	0	0	0
Trees	0	0	0	0	0	1	1	1	0
Graph	0	0	0	0	0	0	1	1	1
Minors	0	0	0	0	0	0	0	1	1

Vector Space Model

- **Documents are represented as vectors in the "term space"**
  - **Terms are usually *stems***
  - **Documents represented by binary or weighted vectors of terms**
- **Queries are represented the same as documents**
- **Query and Document weights are based on length and direction of their vector**
- **A vector distance measure between the query and documents is used to rank retrieved documents**

Documents in 3D Space



Primary assumption of the Vector Space Model:
Documents that are "close together" in space are similar in meaning.

SVD: Text Analytics

LSI: LATENT SEMANTIC INDEXING

Document Space has High Dimensionality

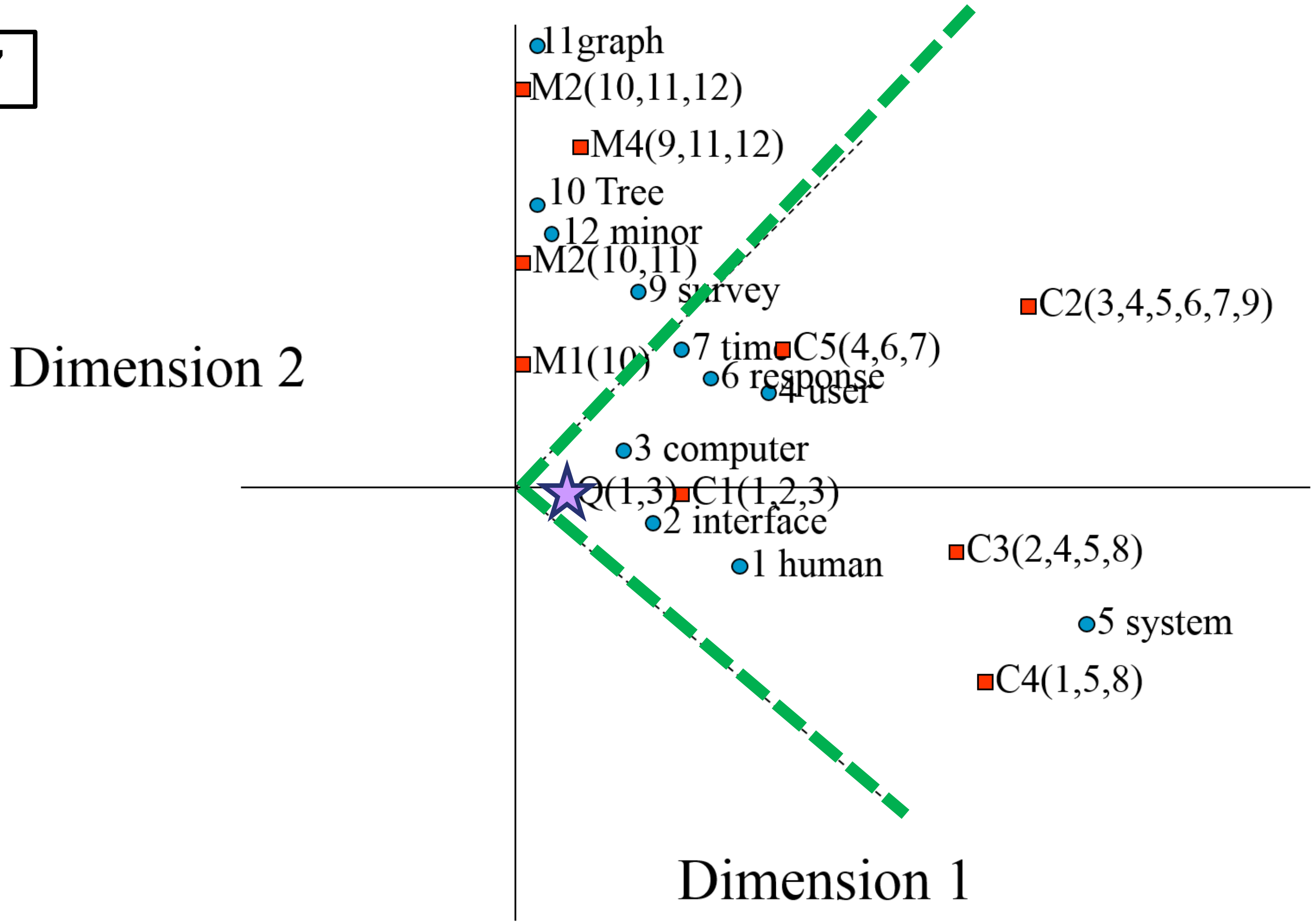
- What happens beyond 2 or 3 dimensions?
- Similarity still has to do with how many tokens are shared in common.
- More terms -> harder to understand which subsets of words are shared among similar documents.
- Approaches to handling (reducing) high dimensionality: Clustering and Latent Semantic Indexing (LSI) (i.e., dimension reduction using SVD)

Rectangular Matrix Decomposition with SVD

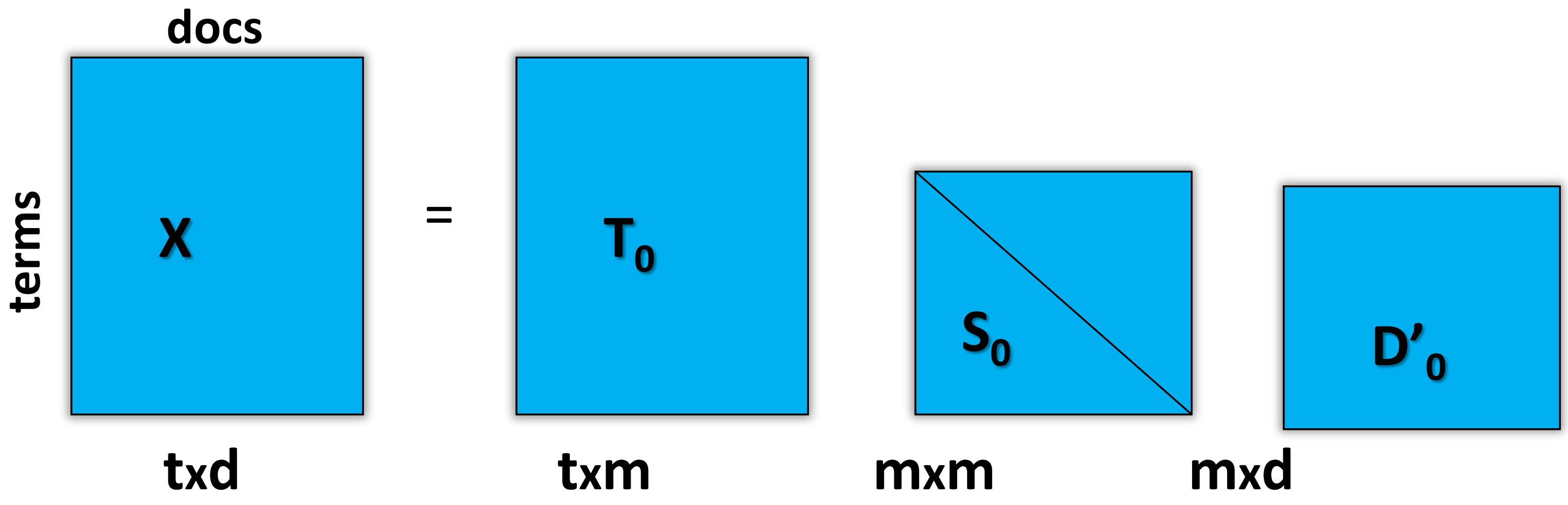
<u>Query</u>: "Human Computer Interaction"

SVD to 2 dimensions:

Blue dots are terms
Documents are red squares
Purple star is a query
Dotted cone is cosine .9 from Query
-- even docs with no terms in common (c3 and c5) lie within cone.



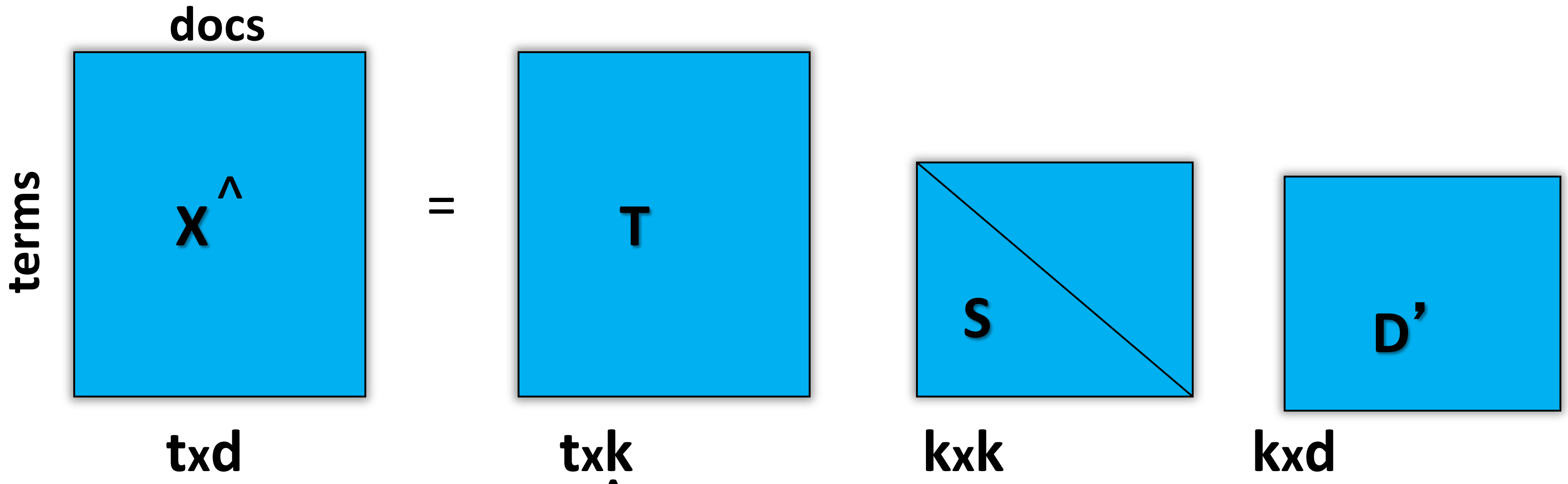
SVD Decomposition of Term-Document Matrix



$$X = T_0 S_0 D'_0$$

T_0 has orthogonal, unit-length columns ($T'_0 T_0 = 1$)
D_0 has orthogonal, unit-length columns ($D'_0 D_0 = 1$)
S_0 is the diagonal matrix of singular values
t is the number of rows in X
d is the number of columns in X
m is the rank of X (<= min(t,d))

Low-Dimensional Approximation of Term-Document Matrix



$$\hat{X} = TSD'$$

T has orthogonal, unit-length columns ($T'T = 1$)
D has orthogonal, unit-length columns ($D'D = 1$)
S_0 is the diagonal matrix of singular values
t is the number of rows in X
d is the number of columns in X
m is the rank of X (<= min(t,d)
k is the chosen number of dimensions in the reduced model (k <= m)

Comparing Two Documents

- The dot product or cosine angle between two column vectors of the matrix X(hat) tells the extent to which two documents have a similar profile of terms/words

Matrices in Graphs / Networks

GRAPHS: MOTIVATION

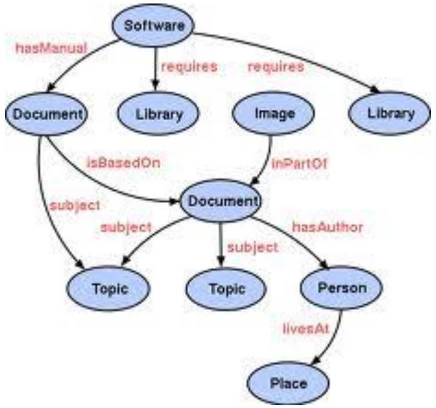
What apps naturally deal w/ graphs?

Social Networks

twitter

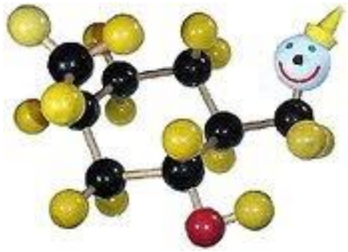
facebook

Semantic Web



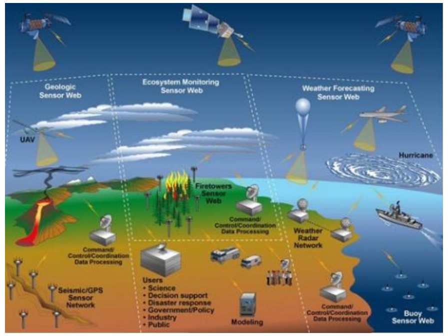
World Wide Web

Drug Design,
Chemical compounds



Computer networks

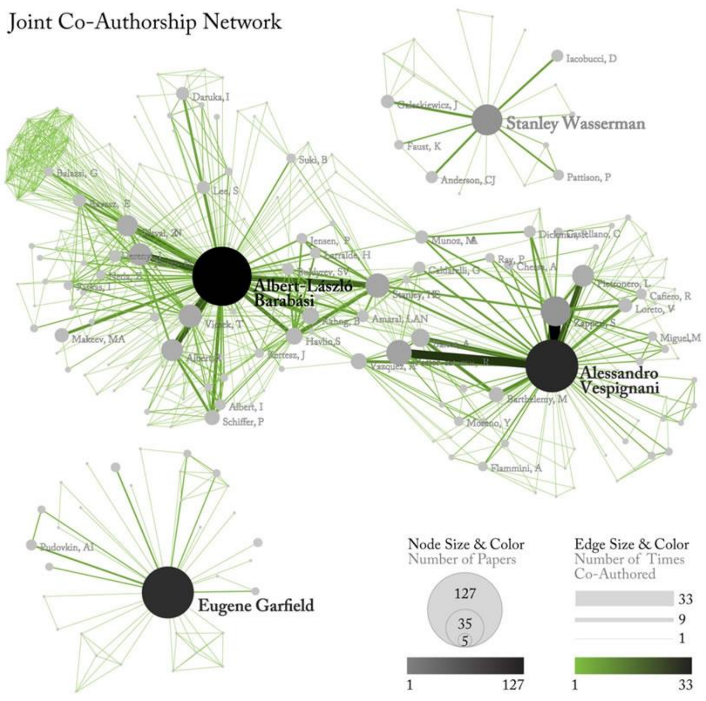
Sensor networks



Credit: Images are from Google images via search of keywords

Real-world Graphs, or Networks

- **Graphs:** consist of objects (vertices) & their relationships (edges/links)
  - **Social:** Facebook, Twitter, Linkedin
  - **Physical:** Internet, power-grid
  - **Biological:** Protein-protein interactions
  - **Academic Collaboration:** Citation, co-author



Can you think of real-world graphs/networks?
- **What are their nodes/vertices?**
- **What are their relationships/edges/links?**

Examples of Real-World Graphs

Graph / Network	Node / Vertex	Edge / Arc / Relationship
Internet	Computer/router	Cable/wireless data connection
WWW	Web page	Hyperlink
Facebook	Person	Friendship
Power Grid	Generating station	Transmission line
Airport Network	Airport in a city	Flight connection
US Roads	City	Roads
Neural Network	Neuron	Synapse

How Big Are These Graphs?

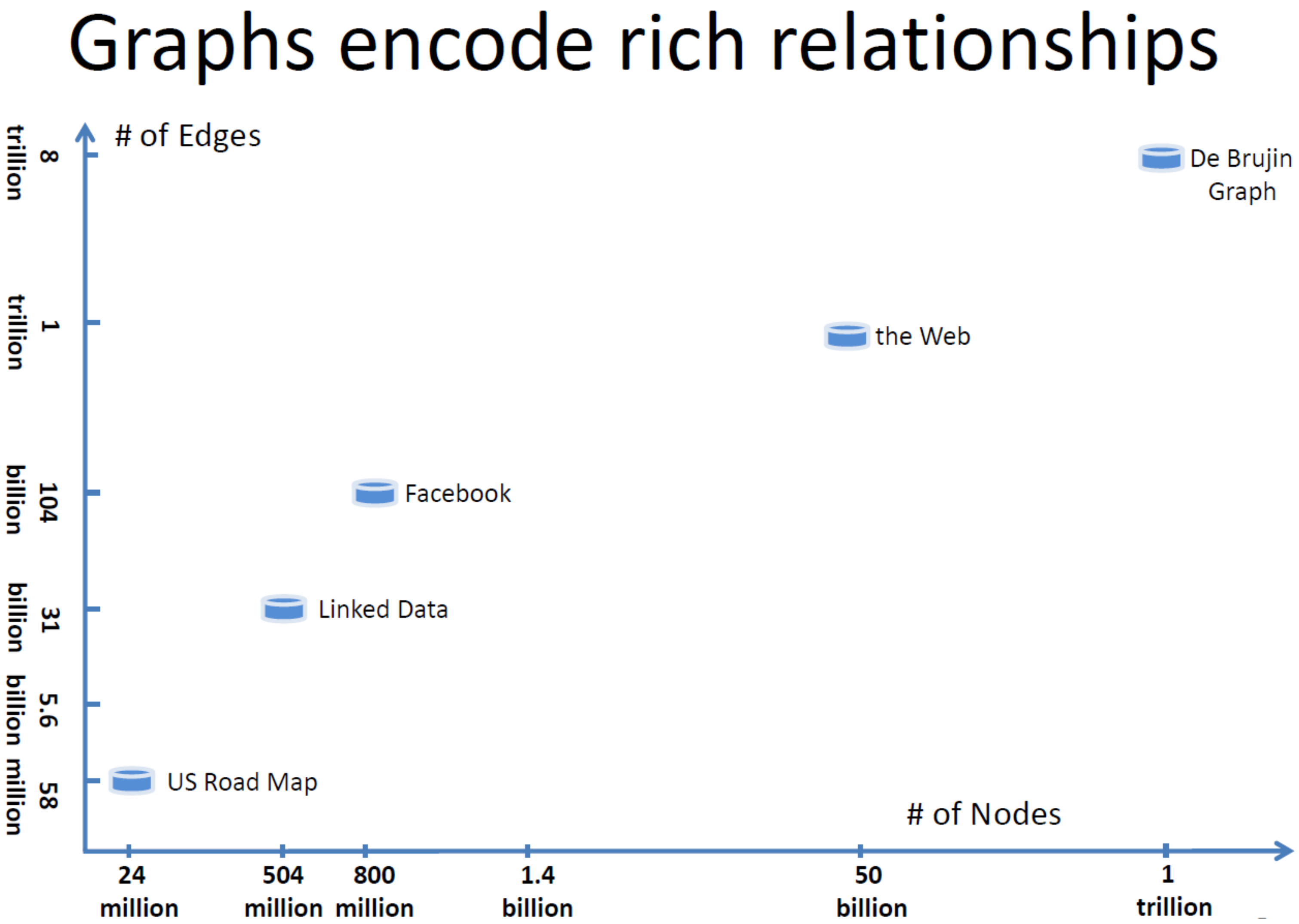


Image source: Haixun Wang, KDD 2012

Graphs Challenge Many Algorithms

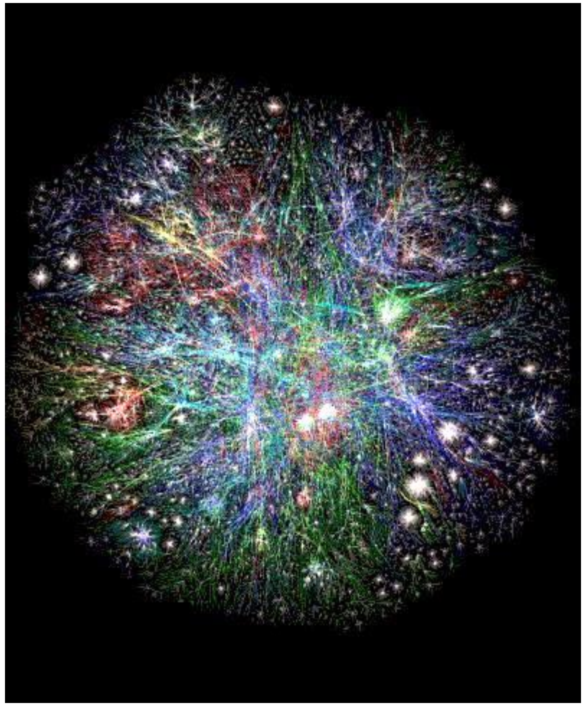
Massive scale

– Can't use conventional algorithms on large graphs

– The graph itself may not even fit in memory

Facebook: |V| = 721M, |E| = 137B
Common crawl: |V| = 3.5B, |E| = 128B

$O(n^2)$ ☹



Matrices in Graphs / Networks

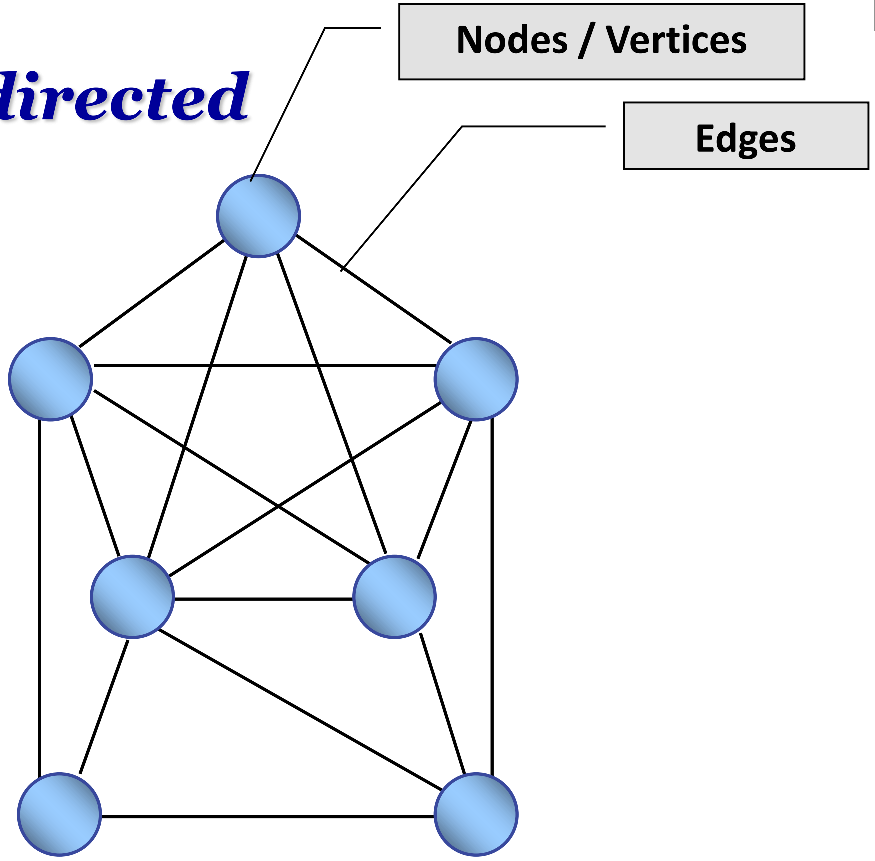
GRAPH THEORY PRIMER

Graphs

Graph with 7 nodes and 16 edges

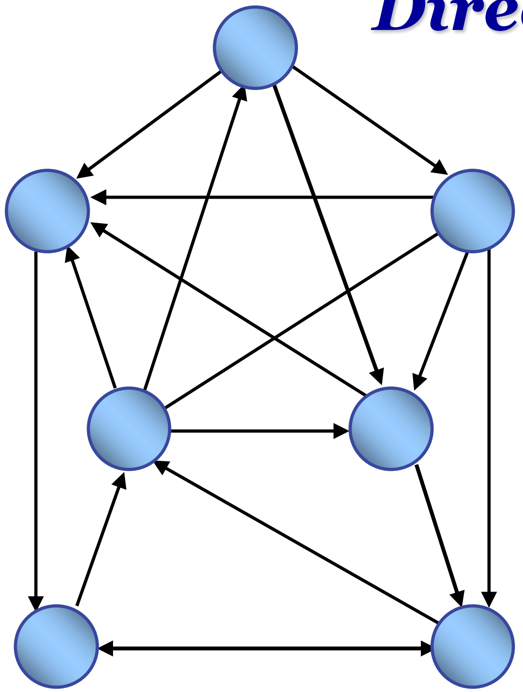
$$G = (V, E)$$
$$V = \{v_1, v_2, \ldots, v_n\}$$
$$E = \{e_k = (v_i, v_j) \mid v_i, v_j \in V, k = 1, \ldots, m\}$$

Undirected



$$(v_i, v_j) = (v_j, v_i)$$

Directed



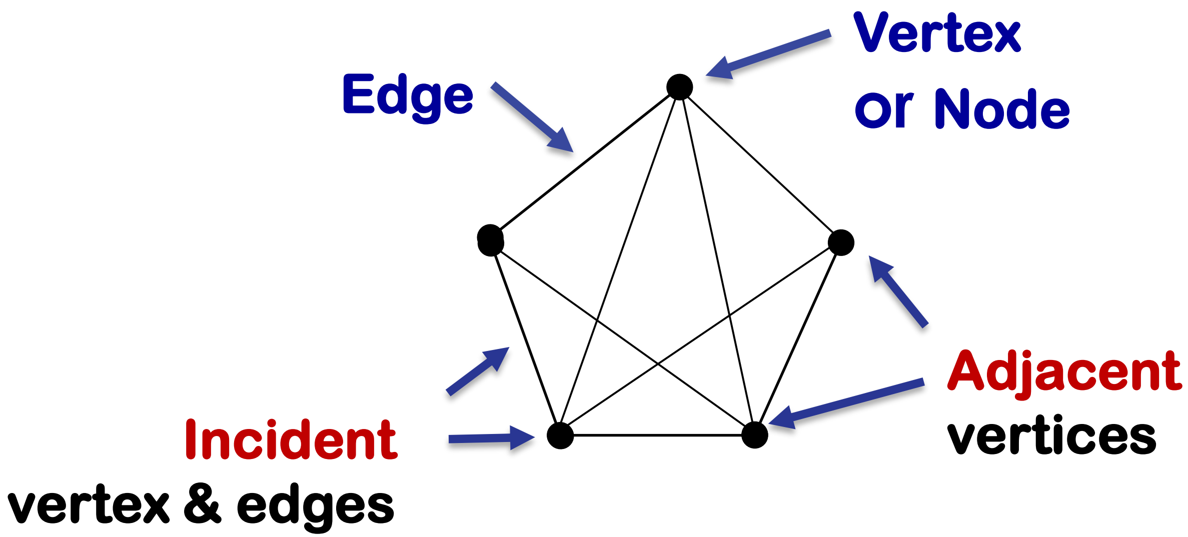
$$(v_i, v_j) \neq (v_j, v_i)$$

Graph Theory

A graph is a collection of *vertices* (nodes) and *edges*.

A vertex represents some object.

An edge connects two vertices and represents some relationship between them.

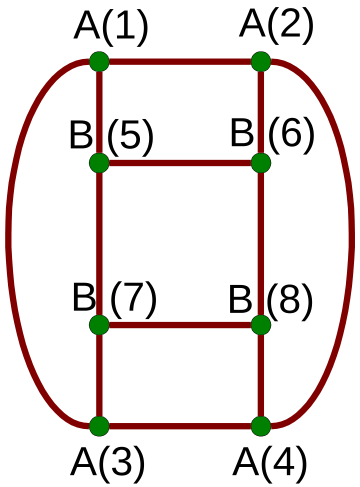


Graph Representation as a Matrix

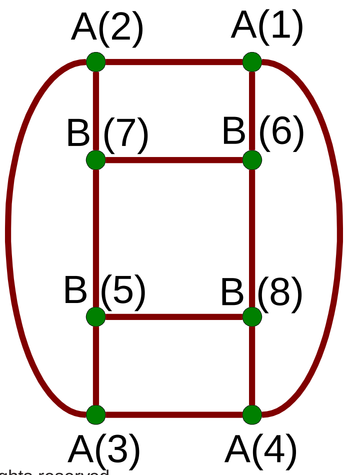
- **Adjacency Matrix** (vertex vs. vertex)
- Incidence Matrix (vertex vs. edge)
- **Sparse** (lots of zero's) vs. **Dense** Matrices

Adjacency Matrix Representation

The representation is *NOT unique*.
Some algorithms are *order-sensitive*.



	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	1	0	1	0	0	0
A(2)	1	1	0	1	0	1	0	0
A(3)	1	0	1	1	0	0	1	0
A(4)	0	1	1	1	0	0	0	1
B(5)	1	0	0	0	1	1	1	0
B(6)	0	1	0	0	1	1	0	1
B(7)	0	0	1	0	1	0	1	1
B(8)	0	0	0	1	0	1	1	1

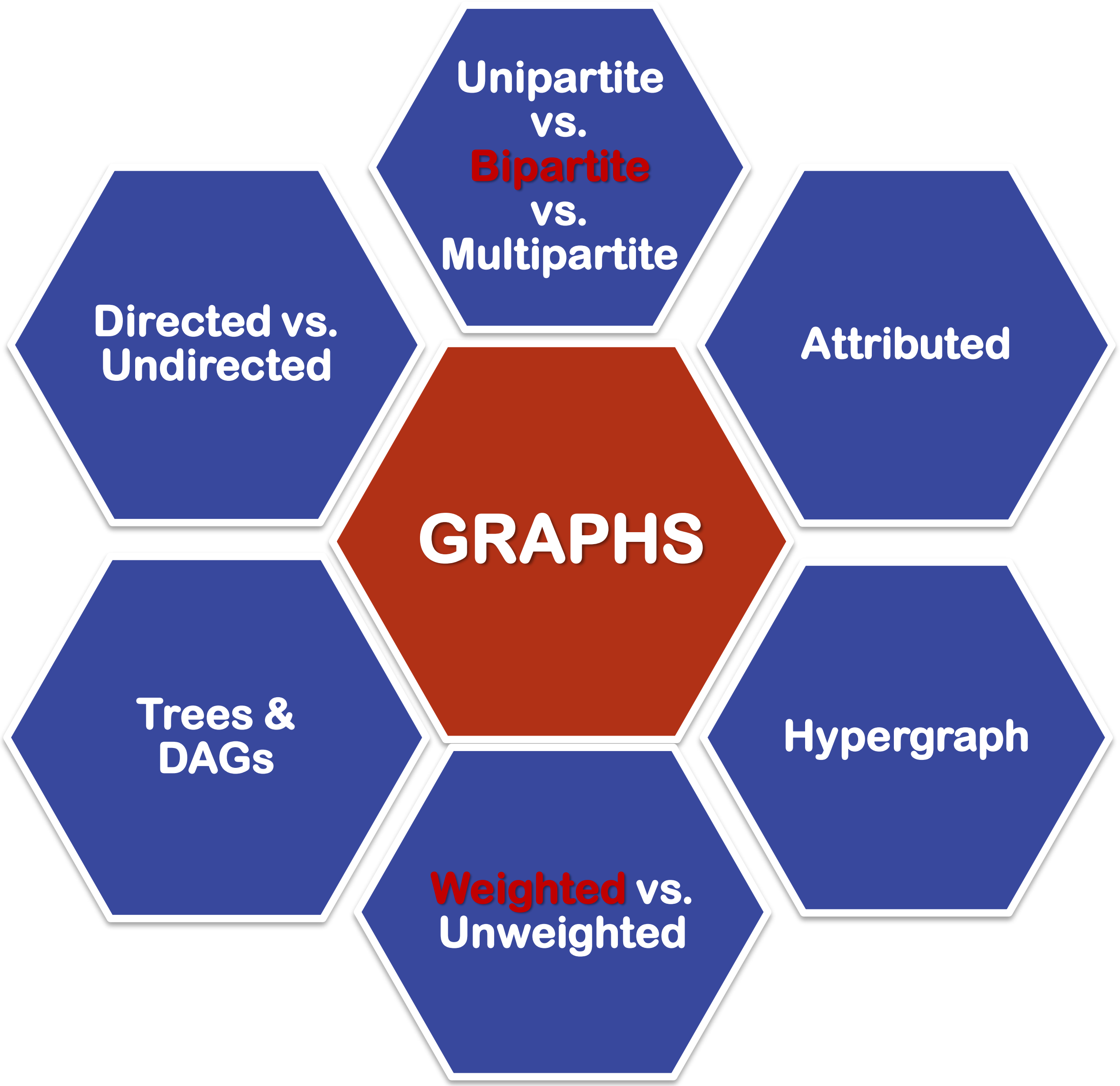


	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	0	1	0	1	0	0
A(2)	1	1	1	0	0	0	1	0
A(3)	0	1	1	1	1	0	0	0
A(4)	1	0	1	1	0	0	0	1
B(5)	0	0	1	0	1	0	1	1
B(6)	1	0	0	0	0	1	1	1
B(7)	0	1	0	0	1	1	1	0
B(8)	0	0	0	1	1	1	0	1

Types of Graphs

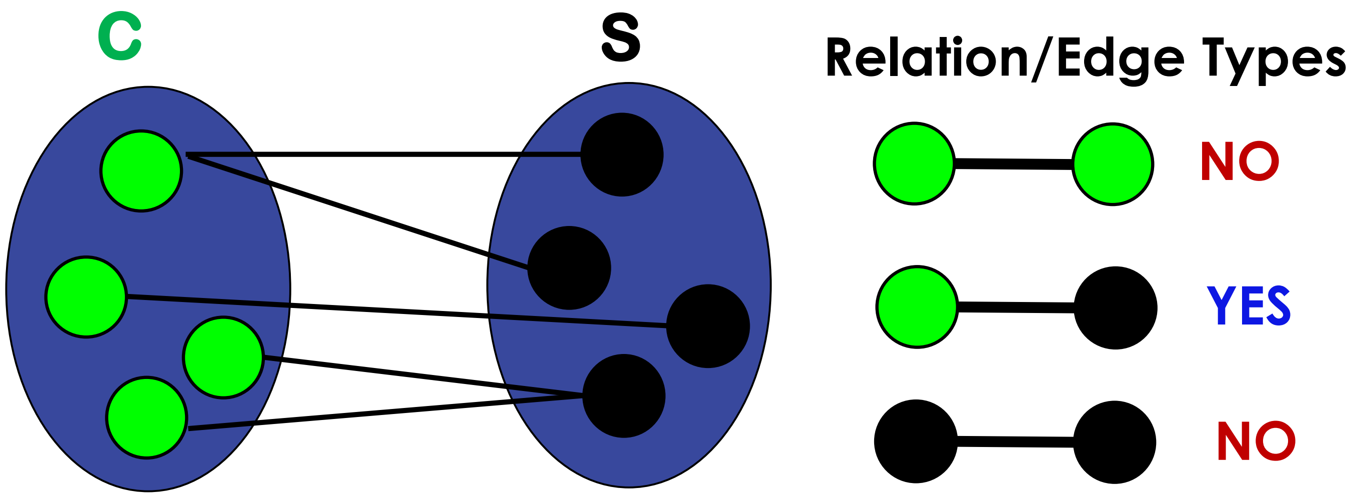
BIPARTITE GRAPHS

Types of Graphs

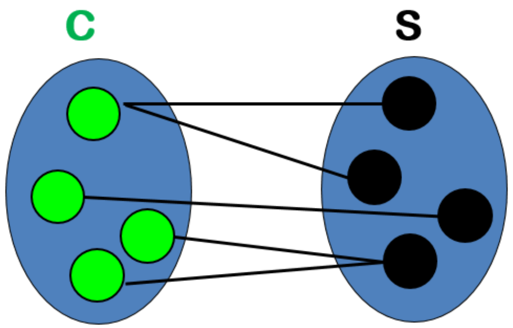


Bipartite Graphs

A graph B is bipartite if its vertices V can be partitioned into two non-intersecting sets C and S such that all relationships/edges go between C and S (no edges go from C to C or from S to S.

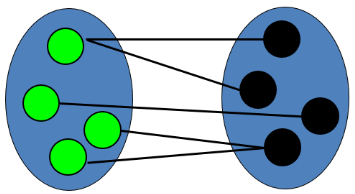


Bipartite Graphs in Business Intelligence (BI)



Business, B	Customers, C	Services, S	Relationships, R
Amazon	Customers	Products	Purchasing
Netflix	Subscribers	Movies	Watching
Pandora	Subscribers	Songs	Listening to
CISCO	Manufacturers	Parts	Supplying to CISCO

Observations about Bipartite BI Graphs



- Adjacency matrix is very **sparse**:($r_{cs} = 0$ or $r_{cs} =???$)
- Relationships are not always binary, but **weighted**:
  - $r_{cs} =*****$: How customer rated a product
  - $r_{cs} = 25$: How many times subscriber listened to the song
  - $r_{cs} = High$: How reliable manufacturer was in supplying the part
- **Missing** relationships ($r_{cs} = 0$ or $r_{cs} =???$) are **business opportunities**:
  - Recommend new products to customers
  - Find reliable manufacturers
  - Ensure continued customer subscriptions to services

Business, B	Customers, C	Services, S	Relationships, R
Amazon	Customers	Products	Purchasing /Rating
Netflix	Subscribers	Movies	Watching
Pandora	Subscribers	Songs	Listening to
CISCO	Manufacturers	Parts	Supplying to CISCO

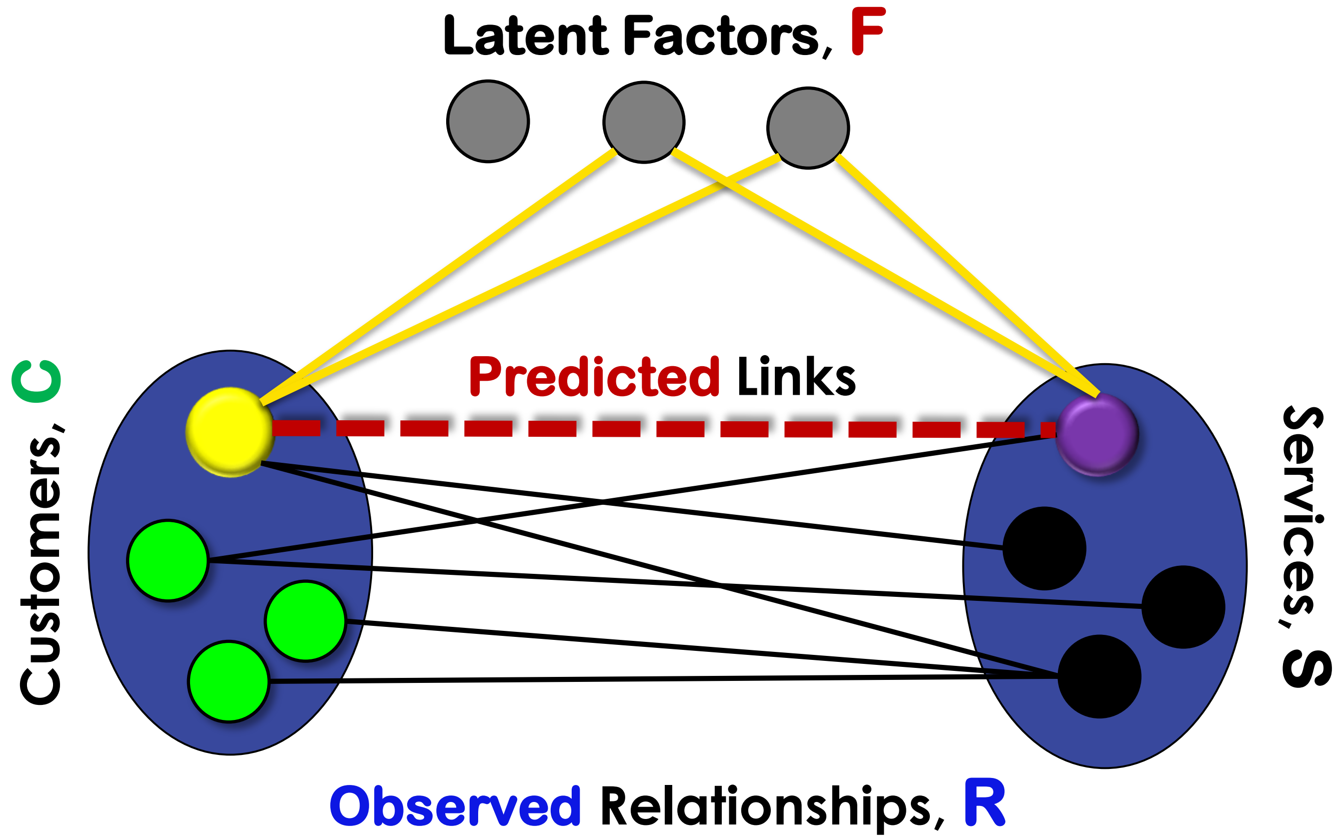
Bipartite Graphs

MISSING LINKS

Assumptions: Hidden Factors & Missing Links

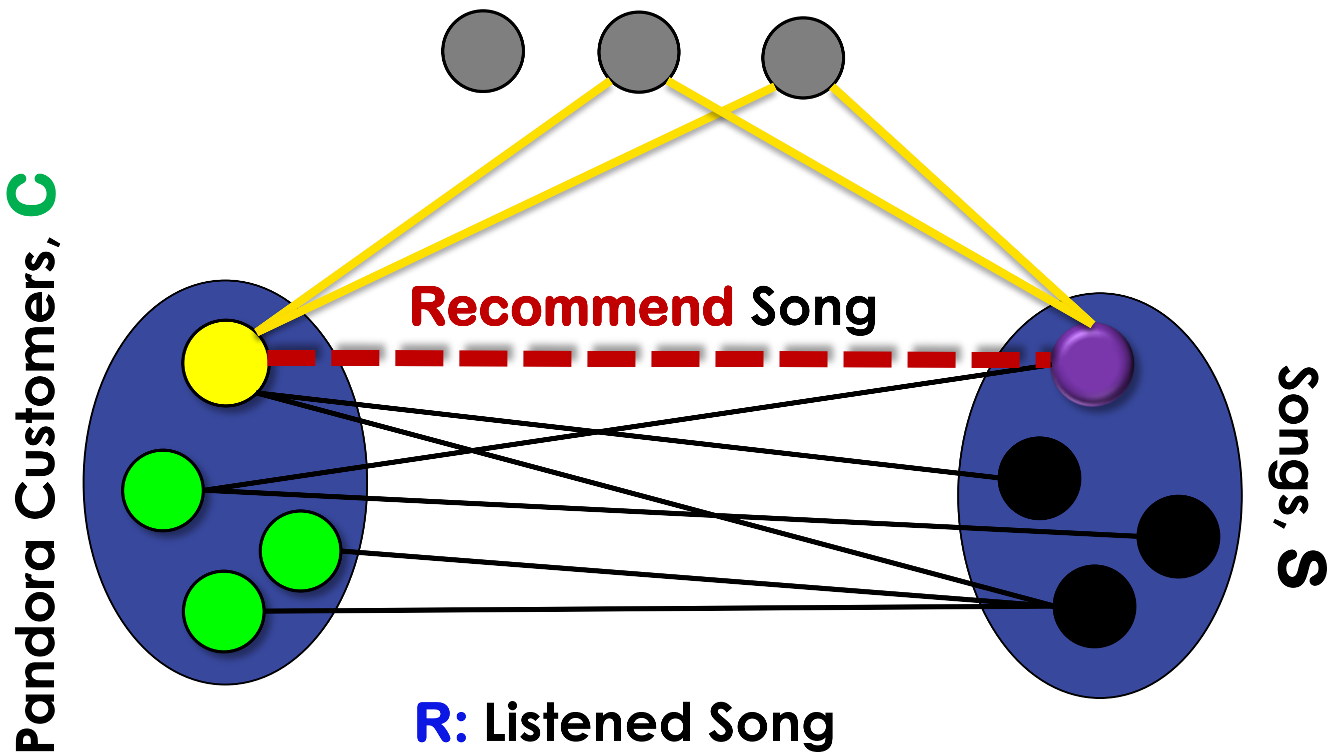
There are a few **hidden** or **latent factors** $F = \{f_1, f_2, \dots, f_h\}$ that define/drive the **relationships** between C and S.

These **latent / hidden factors** can be used **to complete missing relationships** between C and S.



Example: Hidden Factors & Missing Links

Latent Factors, F:
- Tastes
- Genres

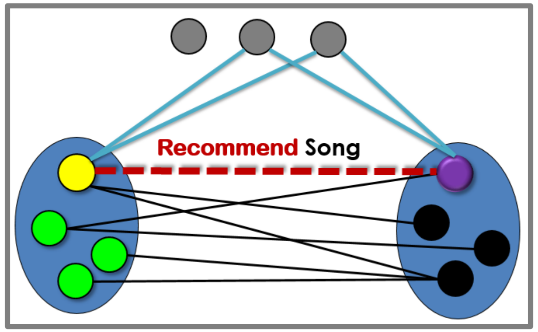


Bipartite Graph Representation

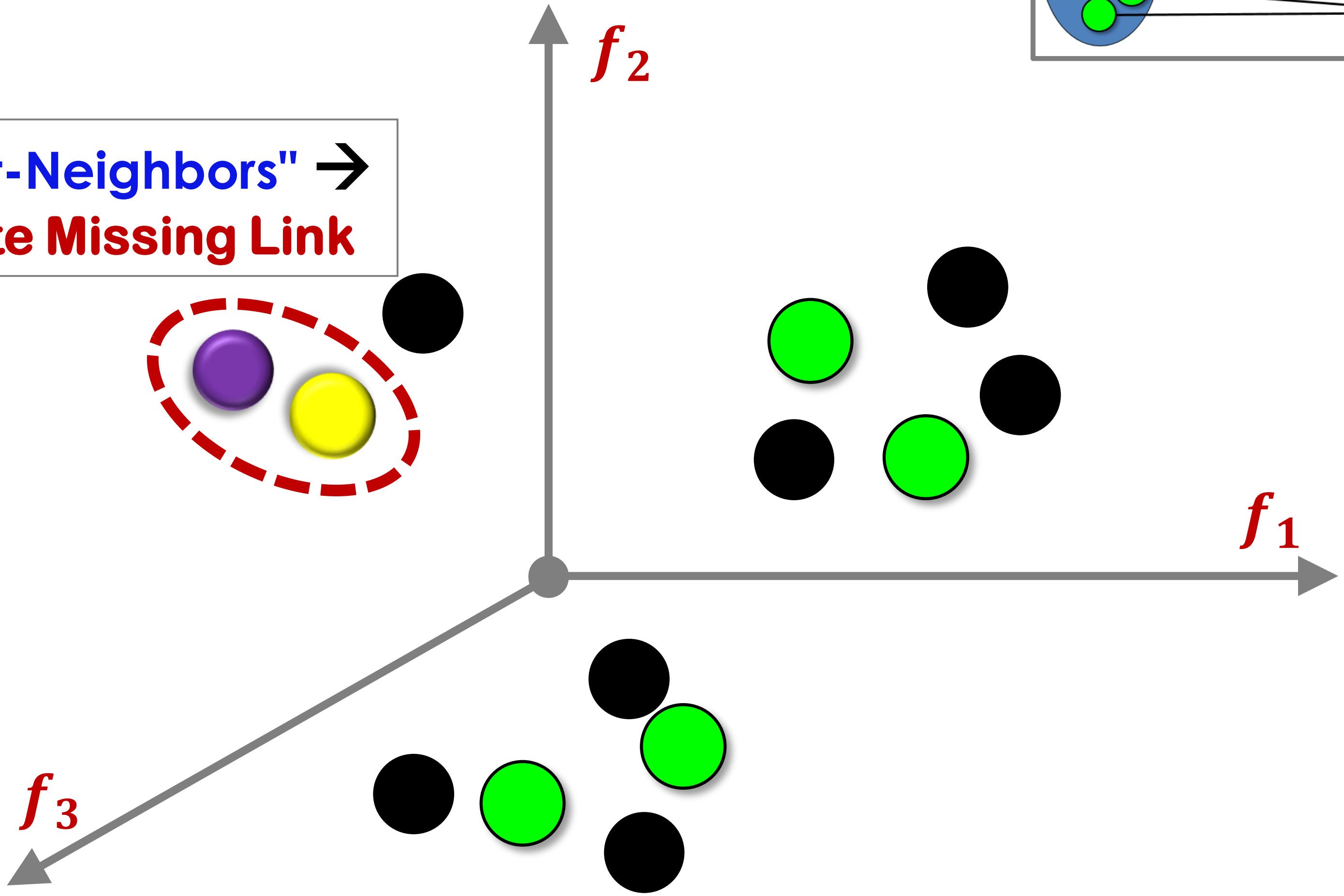
LATENT FACTOR SPACE

Latent Factor Space View

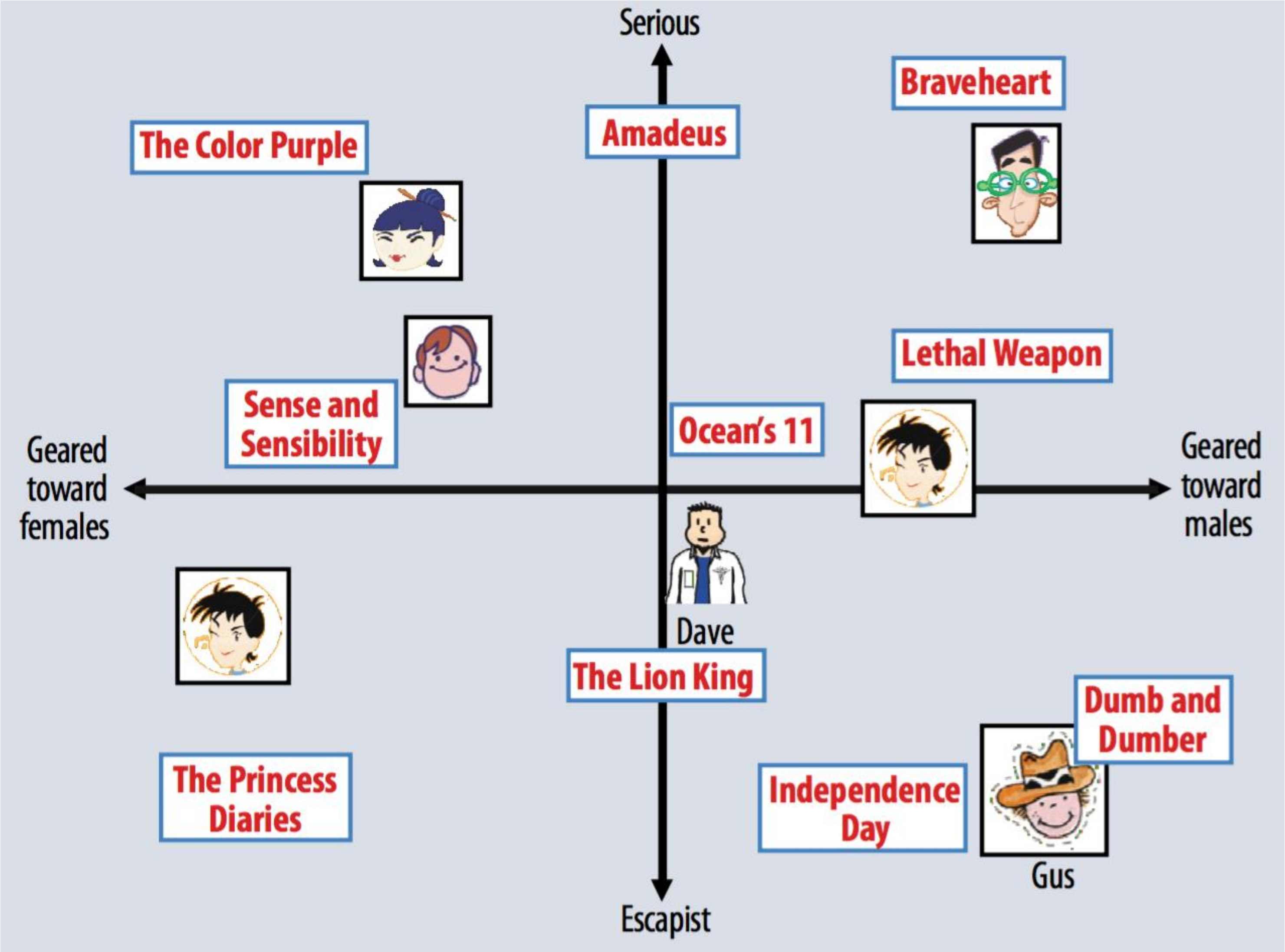
Customer-Service Relations in the Latent Factor Space:
$F = \{f_1, f_2, \ldots, f_h\}$



"Nearest-Neighbors" → Complete Missing Link



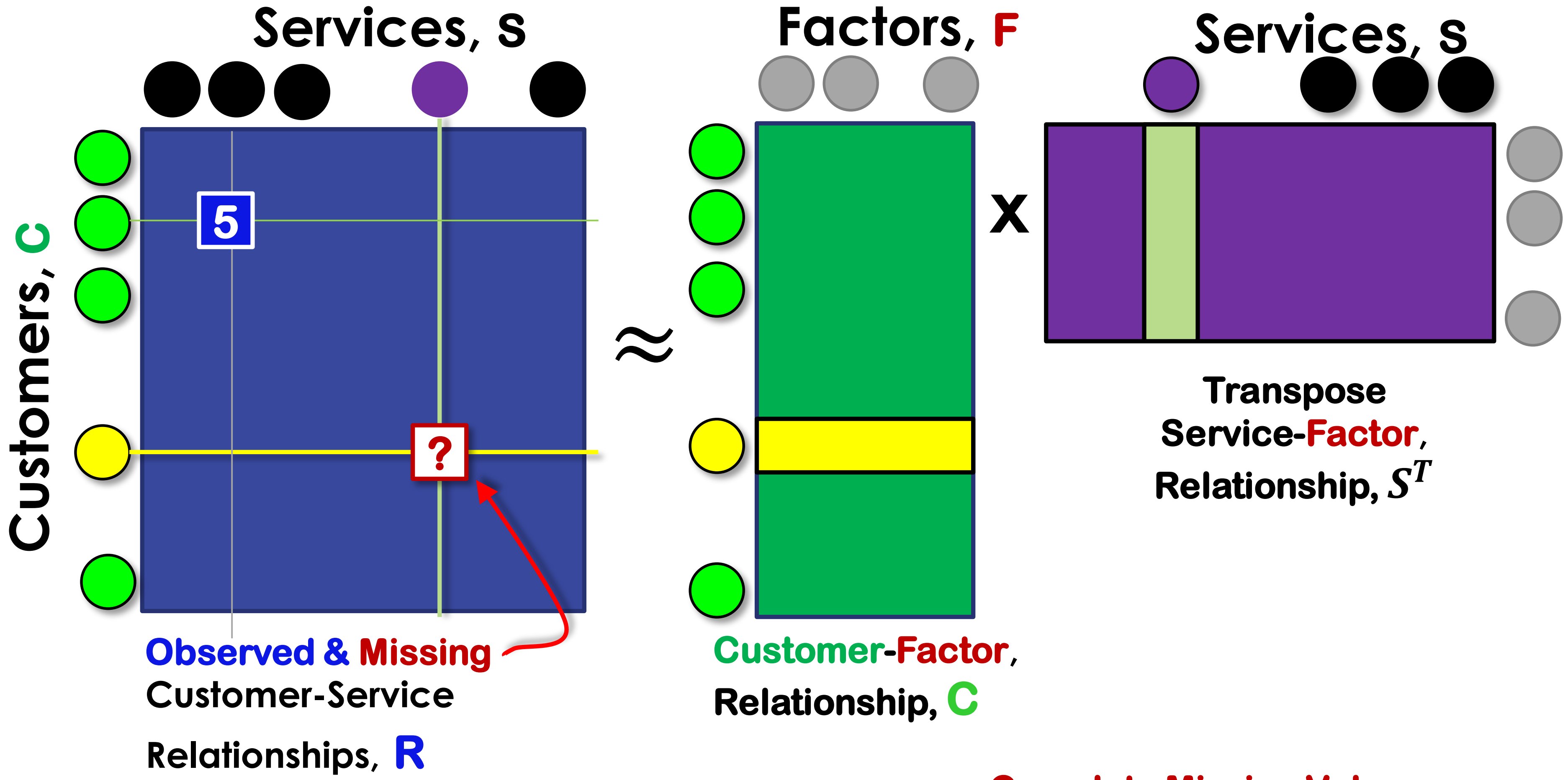
Ex: User-Movie Relations in a Latent Factor Space



Latent Factor Space

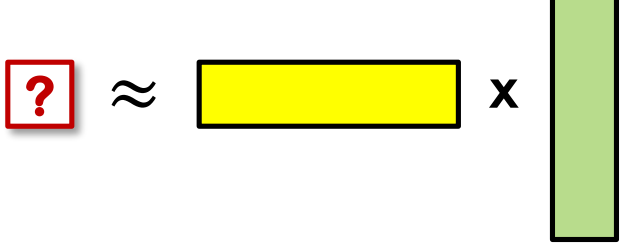
MATRIX FACTORIZATION / DECOMPOSITION

Matrix Factorization / Completion View

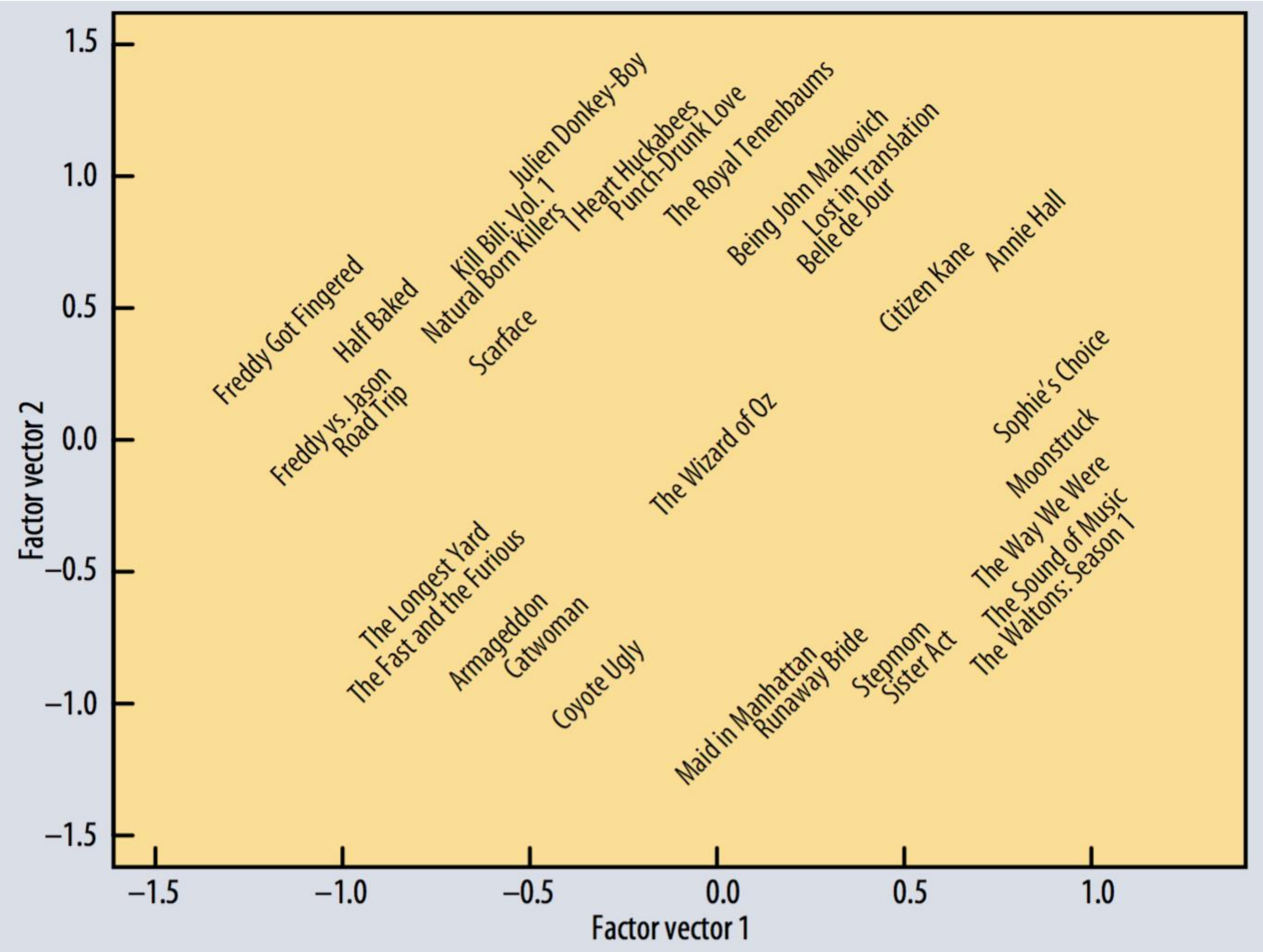


$$\boxed{R \approx C \times S^T}$$

<u>Complete</u> Missing Values



First Two Vectors from Matrix Decomposition

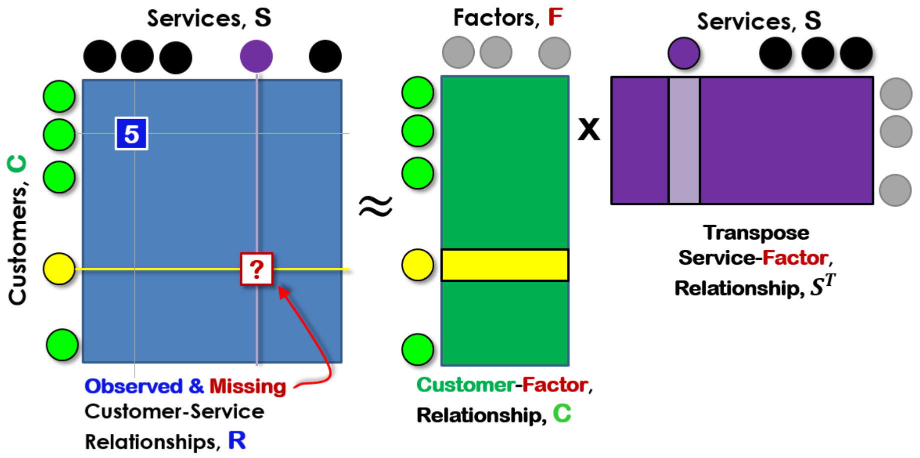


Matrix Factorization / Decomposition

ALTERNATE LEAST SQUARES

Paatero, 1994

Matrix Completion, or Matrix Factorization



Customers: C $= \{c_1, c_2, \dots, c_n\}$

Services: S $= \{s_1, s_2, \dots, s_m\}$

Hidden Factors: $F = \{f_1, f_2, \dots, f_h\}$

$$\boxed{R_{n\times m} \approx \hat{R} = C_{n\times h} \times S^T_{h\times m}}$$

$$\boxed{r_{i,j} \approx \overrightarrow{c_i} \times \overrightarrow{s_j^T}}$$

vector product

R: sparse matrix

$\hat{\text{R}}$: dense matrix

$C \times S^T$: dense matrix

C: dense matrix

S: dense matrix

$h = O(constant)$

$h \ll n$

$h \ll m$

How to Find the "Best" $\hat{R}$?

$$\boxed{R_{n\times m} \approx \hat{R} = C_{n\times h} \times S^T_{h\times m}}$$

known unknown unknown

$$\boxed{r_{i,j} \approx \vec{c_i} \times \vec{s_j^T}}$$

vector product

- **For the sake of simplicity, <u>let's assume</u> that $S^T_{h\times m}$ is known.**
- **How to find $C_{n\times h}$?**
- **Can we multiply on the right by the inverse of $\left(S^T_{h\times m}\right)^{-1}$?**
  - Note: Matrix inverse is only for squared matrices
  - But $S^T_{h\times m}$ is skinny, rectangular ($h \ll m$)
- **How can we create a squared matrix?**

$$\boxed{R_{n\times m} \times S_{m\times h} \approx C_{n\times h} \times (S^T_{h\times m} \times S_{m\times h})}$$

- **$S^T \times S$ is an $h \times h$ squared matrix.**
- **<u>Let's assume</u> that $S^T \times S$ is invertible, then**

$$\boxed{R \times S \times \left(S^T \times S\right)^{-1} \approx C}$$

Assuming S is known and $S^T S$ is invertible

- Goal: Optimize the "best" approximation to the matrix C

$$R \times S \times \left(S^T \times S\right)^{-1} \approx C$$

Minimize Least Squares, or the Frobenius Norm

$$\left\| R \times S \times \left(S^T \times S\right)^{-1} - C \right\|_F \rightarrow min$$

$$E = R \times S \times \left(S^T \times S\right)^{-1} - C$$

Error Matrix, E

$$\|E\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{h} E_{ij}^2} = \sqrt{trace(E^T E)}$$

Let's assume that $C_{n\times h}$ is known

$$\boxed{R_{n\times m} \approx \hat{R} = C_{n\times h} \times S^T_{h\times m}}$$

known known unknown

$$\boxed{r_{i,j} \approx \overrightarrow{c_i} \times \overrightarrow{s_j^T}}$$

vector product

1. How to find $S^T_{h\times m}$?
2. What other assumptions should be made?
3. What is the optimization function?

$$\boxed{???\approx S}$$

$$\boxed{???\rightarrow \textbf{min}}$$

Alternating Least Square

<u>Optimization Problem:</u> Find $R_{n\times m} \approx \hat{R} = C_{n\times h} \times S^T_{h\times m}$

Minimize Loss Function: $\left\|R - C \times S^T\right\|^2_F \rightarrow min$ or equivalently

$$min_{C,S} \sum_{u,i} (r_{ui} - C_u S_i^T)^2$$ ***

- Optimizing C and S simultaneously is non-convex, hard
- If C <u>or</u> S are fixed, then the system of linear equations: convex & easy
  - Initialize S with random values
  - Solve for C
  - Fix C
  - Solve for S
  - Repeat ("Alternating")
    - Fixed number of iterations or
    - Till the Error stabilizes