Topological Matching

Ali Shokoufandeh
Department of Computer Science
Drexel University

Importance of Matching

- Object recognition represents one of the primary goals of computer vision.
- Matching procedure is an important component of object recognition.
- Although some success has been achieved in solving the problem of exemplar-based recognition, we’re a long way from solving the problem of categorical perception.

Matching Problem

- General problem:
  “Given two objects \( A \) and \( B \), we want to know how much they resemble each other”
- In certain settings:
  - One of the objects may undergo certain transformations like translations, rotations or scaling.
- Variations:
  - \( A \) exactly resembles \( B \).
  - \( A \) resembles only to some part of \( B \).
  - A simplified version \( A' \) of \( A \) matches \( B \).

Example:

- Objects:
  - Finite sets of points (“point patterns”)
- Resemblance:
  - Various distance functions, e.g. Hausdorff distance:
    \[
    \delta(A, B) = \max_{a \in A} \min_{b \in B} \| a - b \|
    \]
- Transformation (application dependent):
  - Rigid motions: translation, rotation.
  - Affine: rigid + scaling.
Simple Algorithm for Point Matching

- Given: 2 finite sets $A$, $B$ of $n$ points in $\mathbb{R}^2$.
- Question: are $A$ and $B$ congruent?
- Algorithm:
  1. Determine the centroids $c_A$, $c_B$ of sets $A$ and $B$.
  2. Determine the polar coordinates $(\phi_1, r_1), (\phi_2, r_2), \ldots, (\phi_n, r_n)$ of all points in $A$ using the centroid as the origin, and compute the sequence $S_A = \{ (\gamma_1, r_1), (\gamma_2, r_2), \ldots, (\gamma_n, r_n) \}$, where $\gamma_i = \phi_i - \phi_{i+1} \mod n$.
  3. Compute in the same way the corresponding sequence $S_B$ for $B$.
  4. Determine whether $S_B$ is a cyclic shift of $S_A$, i.e. a substring of $S_A || S_A$ using a fast string matching algorithm.

Observe that

- The problem gets more and more complicated as:
  - Size: Number of points in $A$ and $B$ are not the same.
  - Features: There is additional information associated with each point (in addition to its Euclidean coordinates) in $A$ and $B$.
  - Dimensions: Objects $A$ and $B$ are in spaces of different dimensionality.
  - Objects $A$ and $B$ are not geometric!

Hardness of Categorical Perception

- Categorization requires extracting more abstract (i.e., within-class invariant) features and relations from an image.
- What if we could extract such abstractions?
- Is there a general framework that would allow us to match (recognize) them?

Graph Abstractions and Recognition

- Graphs provide a powerful representational tool for image abstraction, in which:
  - nodes represent image features, and
  - edges represent relations between features.
- Matching two objects can be formulated as matching their respective graph representations.
Example 1: Generic 2-D Shape Matching using Silhouettes

Shock Graph (Siddiqi et. al. 1999)

Example:

Illustrative Example

Computed Correspondence
Where is the Community?

- Combinatorial methods:
  - bipartite matching, tree edit-distance
- Continuous optimization methods
  - maximum clique, graduated assignment
- Algebraic methods
  - spectral methods, matrix/probabilistic methods
- Embedding methods
  - spectral embedding, tree embedding
- Boundaries are blurred - lots of overlap!

Weighted Bipartite Matching

- Known also as assignment problem.
- Given: a bipartite graph $G(A,B,E)$ with weight function $w(e)$ for every $e$ in $E$.

```
A
1 2 3
B
4 5 6
```

- Find: a matching between $A$ and $B$ with greatest total weight.

```
A
0 1 2
B
4 5 6
```

- Note: we may assume $G$ is a complete bipartite graph, with $|A|=|B|$. 

Current Results:

- Bipartite Cardinality Matching:
  - $O(n^{1/2} m)$ time. Max-flow on unit capacity networks. [Even-Tarjan]
- Non-Bipartite Cardinality Matching:
  - First polynomial time, $O(n^3)$, algorithm in 1957 by Edmonds.
  - Current best $O(n^{1/2} m)$ [Micali-Vazirani].
- Bipartite Weighted Matching:
  - $O(nm + n^2 \log n)$ strongly poly.
  - $O(n^{1/2} m \log (nC))$ scaling algorithm.
- Non-Bipartite Weighted Matching:
  - $O(n^3)$ by [Edmonds & Gabow].
  - Current best $O(nm + n^2 \log n)$.
- Where $n=|V|$, $m=|E|$, $C=\max w(e)$.

Application: Bipartite Matching

Title: View-Based Object Recognition Using Saliency Maps
Authors: Marsic, et. al.
Publication: Image and Vision Computing, Vol. 17
Year: 1999
URL: http://www.cs.toronto.edu/~sven/Papers/ivc99.ps.gz

A Coarse-to-Fine Image Representation (Marsic ‘93):

Choosing the Characteristic Scale for an Object
Detecting Peaks at a Given Scale

In 2-D, a set of directional 1-D filters (one of which is shown above) are applied to the image at each position at each scale. Sufficiently large response clusters in a bottom-up search indicate characteristic object scales.

Invariance of the Representation

Illumination Invariance

An Algorithm for Topological Matching

- Adopt a coarse-to-fine solution based on a multilevel bipartite graph matching formulation.
- Two nodes are considered for correspondence if they have no parents at a higher level, or each has a parent at the same level and these parents belong to the matching at that level.
Unoccluded Queries

Results

Match to all other images. Neighbouring pig views were two closest views.

<table>
<thead>
<tr>
<th></th>
<th>Topo</th>
<th>Geo</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>9.57</td>
<td>8.91</td>
</tr>
<tr>
<td></td>
<td>10.06</td>
<td>12.27</td>
</tr>
<tr>
<td></td>
<td>14.58</td>
<td>46.30</td>
</tr>
<tr>
<td></td>
<td>23.25</td>
<td>43.83</td>
</tr>
</tbody>
</table>
Limitations

Representation:

Matching:
- not invariant to vertical compression/expansion of graphs.
- background objects cannot be dominant in the scene.

What is Edit Distance:
- The smallest number of changes (e.g. insertions, deletions, and substitutions) required to change one structure into another.
- It is also referred to as Levenshtein distance.
- Widely used in string matching:
  - The edit distance between two strings is defined by the number of primitive operations (insert, delete, replace) necessary to transform one string to the other.

Example:
- What is the edit distance between “survey” and “surgery”?

```
SURVEY

S U R V E Y

S U R G E Y

S U R G E R Y

Replace (+1)

S U R G E Y

Insert (+1)

S U R G E R Y

Edit Cost (+2)
```
Cost of Edit Operations:

- In the general version of edit distance, different operations may have different costs, or the costs depend on the characters involved.
- For example, replacement could be more expensive than insertion, or replacing “a” with “o” could be less expensive than replacing “a” with “k”.
- The general edit distance does not satisfy the triangular inequality and thus it is not a metric.

Dynamic Programming algorithm

- Problem: find the edit distance between strings \( X \) and \( Y \).
- Create an \((|X|+1)×(|Y|+1)\) matrix \( C \), where \( C_{ij} \) represents the minimum number of operations to match \( X[1…i] \) with \( Y[1…j] \).
- The matrix is constructed as follows:
  - \( C_{0,j} = j \)
  - \( C_{i,0} = i \)
  - \( C_{i,j} \) will be set to
    - \( C_{i-1,j-1} \) if \( X[i] = Y[j] \)
    - \( 1 + \min(C_{i-1,j}, C_{i,j-1}, C_{i-1,j-1}) \) otherwise.

Tree Edit Distance

- Tree edit distance:
  - the minimum cost to transform one tree into another by elementary operations.
- Let \( A \) and \( B \) be ordered trees. We assume
  - The edges are labeled; node labels can be handled similarly.
  - Two kinds of elementary operations are allowed: label modification and edge contraction, with nonnegative costs.
- Goal: find a minimum-cost set of operations to perform on \( A \) and \( B \) to turn them into the same tree.

Example:
Rooted Trees

- We will consider the problem of rooted ordered tree edit-distance.
- Given: two rooted trees $A$ and $B$ where each node’s children are ordered left to right.
- Find: a minimum edit-distance (weighted sequence of edit operations) between $A$ and $B$.

Euler String of a Tree:

- Given an ordered, rooted tree $T$, replace each edge $(x, y)$ by two oppositely directed arcs $(x, y)$ and $(y, x)$.
- The depth-first search traversal of $T$ defines an Euler tour of the darts of $T$.
- We interpret the tour as a string, the Euler string of $T$, denoted by $E(T)$.
- The first dart of the string goes from the root to the leftmost child of the root.

Operations:

- Relation between tree and string edit operations:
  - Contracting an edge in one tree corresponds to deleting a pair of paired parentheses in the corresponding string.
  - Matching an edge in one tree to an edge in the other corresponds to matching the pairs of parentheses and then matching up what is inside one pair against what is inside the other pair.

Current Results:

- The running time of previous algorithm is $O(n^4)$.
- Klein improved this to $O(n^3 \log n)$ for unordered trees through path decomposition and collapsing of trees.
Application: Graph Edit Distance

Title: Recognition of Shapes by Editing Shock Graphs
Authors: Sebastian, Klein, and Kimia
Publication: Proceedings, ICCV 2001
Year: 2001
URL: http://www.lems.brown.edu/vision/publications/conference/SebastianICCV01.ps.gz

Ali Shokoufandeh gratefully acknowledge Thomas Sebastian and Ben Kimia for their contribution of materials to this talk.

Shock Graph Representation of Shape

• Shocks (or medial axis or skeleton) are locus of centers of maximal circles that are bitangent to shape boundary

Edit-Distance for Shock Graphs

• Four edit operations are needed for shock graphs
  – Splice: deletes a shock branch (leaf) and merges the remaining two
  – Contract: deletes a shock branch between two degree-three nodes
  – Merge: combines two branches at a degree-two node
  – Deform: changes the attributes of a shock branch

Real Example
**Edit Cost**

- Edit cost is limit of deform cost:
  - for splice, cost of deforming protrusion to circular arc (see below)
  - for contract, cost of deforming edge to a point
  - for merge, topology is unaffected, so cost reduces to deformation cost.

**Edit-Distance Between Shapes**

- Edit-distance is defined as the sum of the cost of edits in optimal edit sequence

**Matching Examples**

Edit-distance algorithm gives intuitive results

Note that same colors indicate matching edges; gray-colored edges are pruned

**Boundary Noise**

- Shock graphs are sensitive to boundary noise

In optimal edit sequence, “noisy” branches are pruned
**Viewpoint Variation**

- **Smooth change**
- **Abrupt change**

- Deform edit handles smooth changes
- Splice and contract edits handle abrupt changes

**Partial Occlusion**

- Edit distance is robust to partial occlusion

**Generic Recognition**

- Results duplicated in two databases: 99 shapes and 216 shapes

**Conclusions**

- Novel application of graph edit distance to yield an elegant shape matching framework using shock graphs.
- Handles graph regularization within the matching algorithm, making it applicable to a wide variety of domains.
- Both discrete (graph) and continuous (curve) deformation costs are mapped to a continuous curve deformation cost, providing a metric.
- Outstanding results, reflecting strong invariant properties.
Distance Matrix between Objects:

Assume we are given two objects \( A = \{a_1, \ldots, a_m\} \) and \( B = \{b_1, \ldots, b_n\} \) and a distance matrix that represents the similarity between any two objects in \( A \) and \( B \).

Without loss of generality we may assume:
- \( |A| = |B| \).
- \( D_{ij} \geq 0 \) and \( D_{ii} = 0 \).
- \( D_{ij} = D_{ji} \).
- \( D \) is a symmetric nonnegative definite matrix.

Stochastic Matrices:

- A permutation matrix has all entries equal zero except one entry 1 in each row and each column.
- Equivalently, it is a matrix defined by a permutation \( \sigma \in S_n \) as
  \[ P_\sigma = (\delta_{\sigma(i),j}) \]
- A doubly stochastic matrix is a nonnegative matrix whose rows and columns each add up to 1:
  \[ a_{ij} \geq 0, \sum_j a_{ij} = 1, \sum_i a_{ij} = 1. \]

Matrix Scaling Problem:

- An \( n \times n \) matrix \( A \) is said to be scalable if there exist two diagonal matrices \( X \) and \( Y \) such that \( XAY \) is double stochastic.
- The scaling problem is to determine the scalability of a given matrix, and to find the scaling factors.
- Algebraically, the problem can be stated as the system of nonlinear equations in positive variables:
  \[ A^T x = y^{-1}, \quad Ay = x^{-1}, \]
  \[ x > 0, \quad y > 0, \]
- Where \( x^{-1} = (1/x_1, \ldots, 1/x_n) \) and \( y^{-1} = (1/y_1, \ldots, 1/y_n) \).

Note:

- Any solution to matrix scaling problem is an stationary point of the so called logarithmic barrier function [Marshall & Olkin (1968)]:
  \[ g(x, y) = x^T Ay - \sum_i \ln x_i - \sum_i \ln y_i. \]
- For a fixed \( y \), the function \( g(x, y) \) is convex in \( x \). We can analytically minimize \( g(x, y) \) over \( x > 0 \) to get :
  \[ x(y) = \arg \min g(x, y) = (Ay)^{-1}. \]
- We arrive at coordinate-decent, row and column normalization method:
  \[ x_{k+1} = (Ay_k)^{-1}, \quad y_{k+1} = (A^T x_{k+1})^{-1}. \]
- The method is known to converge for scalable matrices [Sinkhorn (1967)].
An Equivalent Formulation:
- The matching problem can be also formulated as a linear programming problem:

$$\max \sum_{(u,v) \in E(G)} W(u,v) X(u,v)$$

Subject to:
$$\sum_{(u,v) \in E(G)} X(u,v) \leq 1, \forall u \in V(G), \text{(I)}$$
$$X(u,v) \in [0,1], \forall (u,v) \in E(G), \text{(II)}.$$  

Where $X(u,v)$ is the weight of edge $(u,v)$.

- The constraints (II) will force the selection or rejection of every edge.
- The constraints (I) will guarantee that for every vertex only one of its incident edges is selected.

Graph Isomorphism Problem:
- Given: two graphs $G=(V(G),E(G))$ and $H=(V(H),E(H))$.
- Problem description: Find a bijective mappings of the vertices of $G$ to the vertices of $H$ such that $G$ and $H$ are identical; i.e., $(x,y)$ is an edge of $G$ iff $(f(x),f(y))$ is an edge of $H$.
- This also referred to as four vertex constraint.
- We will write $G \cong H$ if $G$ and $H$ are isomorphic.
- Maximum Subgraph Isomorphism:
  - Find maximum sub-graphs $G' \subseteq G$ and $H' \subseteq H$, such that $G' \cong H'$.
- The problem is known to be hard for NP.

Matrix Form
- Let us define the isomorphism matrix $M$ between graphs $G$ and $H$ as:

$$M_{uv} = \begin{cases} 
1 & \text{if } u \in V(G), v \in V(H) \text{ and } v = f(u) \\
0 & \text{Otherwise} 
\end{cases}$$

- Observe that similar constraints to ILP formulation is true:

$$\sum_{v \in V(H)} M_{uv} \leq 1, \forall u \in V(G),$$
$$\sum_{u \in V(G)} M_{uv} \leq 1, \forall v \in V(H).$$

Quadratic Formulation:
- The four vertex constraint will form the basis of the optimization formulation of maximum graph isomorphism problem:
Quadratic Formulation (cont’d.):

- These four cases can be captured via a rectangular term $M_{u,s} \times M_{v,t} \times C(u,v), (s,t)$, for all $u,v \in G$ and $s,t \in H$, where the term $C(u,v), (s,t)$ is the measure of compatibility between edges $(u,v)$ or $(s,t)$ and will be defined as:

$$C(u,v), (s,t) = 1 - \frac{1}{3} |A^u_{u,v} - A^v_{s,t}|$$

- Where for a graph $A^G$, and $A^H$ represent the adjacency matrices of graph $G$ and $H$.

- Observe that:

$$H \neq G \quad \text{then} \quad \forall (u,v), (s,t)$$

Note:

- If under an isomeric mapping $f$, $s=f(u)$ and $t=f(v)$ then $M_{u,s} \times M_{v,t} \times C(u,v), (s,t) = 1$, otherwise the rectangular term will have value $0$.

- [Gary and Johnson (1979)] The maximum isomorphism problem then can be restated as finding a symmetric $\{0,1\}$ matrix $M$ that maximizes:

$$\sum_{u,v \in G} \sum_{s,t \in H} \sum_{u',v' \in G} \sum_{s',t' \in H} M_{u,u'} M_{v,v'} C(u,v), (s,t)$$

subject to:

- $\sum_{u \in U} M_{u,s} \leq 1, \forall (u,v) \in V(G)$,
- $\sum_{v \in V} M_{s,v} \leq 1, \forall (u,v) \in V(H)$,
- $M_{u,s} \in \{0,1\}, \forall u \in V(G), s \in V(H)$.

A Graduated Assignment Algorithm for Graph Matching

**Title:** A Graduated Assignment Algorithm for Graph Matching

**Authors:** Gold and Rangarajan

**Publication:** IEEE PAMI, Volume 18, Number 4, April 1996, pp 377-388.

**URL:** http://noodle.med.yale.edu/~anand/ps/pamigm3.ps.gz

Weighted Graph Matching

For weighted graphs, $G$ and $g$ (whose links may take values in $\mathbb{R}$), find the match matrix $M$ which minimizes:

$$E_wg(M) = \frac{1}{2} \sum_{a=1}^{A} \sum_{i=1}^{l} \sum_{b=1}^{A} \sum_{j=1}^{l} M_{ai} M_{bj} C_{abij}$$

subject to:

$$\forall a \sum_{i=1}^{l} M_{ai} \leq 1, \forall i \sum_{a=1}^{A} M_{ai} \leq 1, \forall ai M_{ai} \in 0, 1$$
**Adding a Second Constraint**

1. Recall that a node in graph $G$ may correspond to only one node in graph $g$.
2. But, the reverse also holds: a node in graph $g$ may correspond to only one node in $G$.
3. How do we then ensure that in addition to only one entry per row converging to one, only one entry per column converges to one?
4. Sinkhorn proved that any square matrix with positive elements will converge to a doubly stochastic matrix through an iterative process that alternatively normalizes the rows and columns.

---

**The Assignment Problem**

Back to our maximization sub-problem.

Given $\{X_i\}, X_i \in R^1, M_{i,j} \in \{0,1\}, 8 \sum_i / M_{i,j} = 1$ and $8 \sum_j / M_{i,j}$, our goal is to find $M$ (a permutation matrix) that maximizes:

$$E_{opt}(M) = \sum_{i=1}^{4} \sum_{j=1}^{l} M_{i,j} X_{i,j}$$

This is known as the assignment problem in combinatorial optimization.

---

**Algorithm**

Initialize $\beta$ to $\beta_0$

Begin A: (Do A until $(\beta, \beta_f)$)

- $M_{i,j} \propto e^{\beta_i}$

Begin B: (Do B until $M$ converges)

- Update $M$ by normalizing across all rows
  $$M_{i,j} \propto M_{i,j} / \sum_i M_{i,j}$$

- Update $M$ by normalizing across all columns
  $$M_{i,j} \propto M_{i,j} / \sum_j M_{i,j}$$

End B

End A
Back to Weighted Graph Matching

1. Although the assignment problem can be solved in polynomial time, our weighted graph matching problem is a quadratic assignment problem, which is NP-complete:

   \[ E_{\text{QP}}(M) = \frac{1}{2} \sum_{a=1}^{A} \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} M_{ai} M_{kj} C_{ijk} \]

2. We will find an approximate solution to our quadratic assignment problem by using a continuation method to solve a succession of assignment problems.

3. For each assignment problem, a globally optimal, doubly stochastic matrix is returned for the current value of the control variable

Approach

1. start with an initial \( M \)
2. compute first-order Taylor series expansion
3. find the softassign for the current assignment
4. substitute the resulting \( M \) back into the original energy formulation
5. slowly increase control parameter \( \beta \) until the algorithm “pushes” \( M \) toward an integer solution.
6. outliers and/or missing nodes can be accommodated by introducing slack variables (adds an extra row and column to \( M \)), turning the inequality constraints into equality constraints

Demonstration: Image Feature Graphs

- image of coffee cup with features hand labeled
- image of table top with features hand labeled

Experiments: Random Graphs

- graph size = 100, trials per plot = 700, plots are % of nodes deleted.
Experiments: Weighted Graph Matching

Graph size = 100, trials per plot = 600, plots are % of nodes deleted. Left: no deleted or spurious links. Right: links 5% spurious, 5% deleted. (a) 40% deleted, 15% connectivity; (b) 40%, 10%; (c) 60%, 15%; (d) 60%, 10%.

Limitations

1. GA can handle missing nodes, added/deleted edges, and perturbed attribute values, but not spurious nodes that affect existing graph structure.
2. Does not support many-to-many matching.
3. For object recognition, requires a linear search of the database - no indexing support.

Matching Trees

- Let \( T_1(V_1,E_1) \) and \( T_2(V_2,E_2) \) denote two trees and consider the problem of maximum isomorphism between \( T_1 \) and \( T_2 \):
  - Find a maximum subsets of vertices \( V'_1 \subseteq V_1 \) and \( V'_2 \subseteq V_2 \) \( |V'_1|=|V'_2| \), and a mapping \( \varphi : V'_1 \rightarrow V'_2 \) such that, for all \( u,v \in V_1 \), \( (u,v) \in E_1 \) if and only if \( (\varphi(u),\varphi(v)) \in E_2 \).

- We will transform this problem to that of computing a maximum clique in a related graph.

Product Graph

- Given two graphs \( H \) and \( H' \) their product graph (association graph) \( G=G_{H,H'} \) is defined as follows:
  - \( V(G)=V(H)\times V(H') \).

- Let \( u,v \in V(H) \) and \( x,y \in V(H') \), and let \( \alpha=[u,x] \) and \( \beta=[v,y] \) two of the possible vertices in \( G \), then \( (\alpha,\beta) \) is an edge in \( G \) if \( d(u,v)=d(x,y) \), where \( d(.,.) \) is a distance function in corresponding graphs.
Isomorphism and Cliques:

- **Theorem [Pelillo(2002)]:** Any maximal (maximum) subtree isomorphism between two trees induces a maximal (maximum) clique in the corresponding product graph and vice versa.

Prelude to Optimization

- Let $A=A(G)$ denote the $n\times n$ symmetric adjacency matrix of graph $G$:
  
  $A_{x,v} = \begin{cases} 1 & \text{if } (u,v) \in E(G), \\ 0 & \text{Otherwise.} \end{cases}$

- Define the quadratic form $f(x)$ for $x \in \mathbb{R}^n$ as:
  
  $$f(x) = \frac{1}{2} x^T A x = \frac{1}{2} \sum_{x_i, x_j \in E(G)} A_{x_i, x_j} x_i x_j.$$  

  We have that:
  
  $$f(x) = \frac{1}{2} x^T A x = \sum_{x_i, x_j \in E(G)} x_i x_j.$$  

Quadratics Optimization

- Consider the following quadratic programming problem:
  
  $$\max \quad f(x) = \frac{1}{2} x^T A x = \sum_{x_i, x_j \in E(G)} x_i x_j$$

  S.T. \quad $\sum_{i=1}^{n} x_i = 1$

  $x_i \geq 0, \forall i = 1, \ldots, n$

  - The constraints of this problem will identify the $n$ dimensional simplex.

  $$S_n = \left\{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} x_i = 1, x_i \geq 0, \forall i = 1, \ldots, n \right\}$$

  $S_n$ is a simplex in $n$ dimensions.

Note:

- The expression for $f$ is increasing in $k$, and so the best we can do is when $k$ is cardinality of the maximum clique in the graph $G=(V,E)$, i.e.,

  $$k=g(G)=\alpha(G).$$

- **Theorem [Motzkin-Strauss]:**
  
  $$\max_{x \in R^n, x_i \geq 0} \frac{1}{2} x^T A x = \frac{1}{2} \left( 1 - \frac{1}{\gamma(G)} \right).$$
Subgraph Isomorphism as a Maximum Clique Problem

\[(u,w) \sim (v,z) \text{ iff } (u \sim v \text{ AND } w \sim z) \text{ OR } (u \not\sim v \text{ AND } w \not\sim z)\]

Theorem 2

Let \( C \subseteq V \) and let \( X_C \) be its characteristic vector. Then:

- \( C \) is a maximum clique of \( G \), \( X_C \) is a global maximizer of \( g \) in \( S_n \). In this case, \(|C| = \frac{1}{2}(1 - g(X_C))\).
- \( C \) is a maximal clique of \( G \), \( X_C \) is a local maximizer of \( g \) in \( S_n \).
- All local (and hence global) maximizers of \( g \) on \( S_n \) are strict.

See (Bomze, Pelillo, and Stix, 1999) for proof.

A Stronger Result

Bomze has recently introduced a solution using a regularized version of \( f \):

\[g(x) = x^T A x + \frac{1}{2} x^T I_n\]

which is obtained by substituting in \( f \) the following adjacency matrix:

\[\hat{A}(x) = A + \frac{1}{2} I_n\]

where \( I_n \) is the \( n \times n \) identity matrix.

Solution

Bomze has recently introduced a solution using a regularized version of \( f \):

\[g(x) = x^T A x + \frac{1}{2} x^T I_n\]

which is obtained by substituting in \( f \) the following adjacency matrix:

\[\hat{A}(x) = A + \frac{1}{2} I_n\]

where \( I_n \) is the \( n \times n \) identity matrix.
If $f$ is linear, i.e., $f$ is an $n \times n$ matrix such that $f_i(x) = (Wx)_i$, then we can write:

$$
\dot{x}_i(t) = x_i(t)(f_i(x(t)) - f(x(t))), \quad i = 1, \ldots, n
$$

$$
\ddot{x}_i(t) = x_i(t)\left[(Wx(t)), -x(t)^TWx(t)\right], \quad i = 1, \ldots, n
$$

and in the discrete domain:

$$
x_i(t+1) = x_i(t) - x_i(t)(Wx(t)), \quad i = 1, \ldots, n
$$
Matching Problem Formulation

**Challenge:**
Due to noise and occlusion, significant isomorphic subgraphs may simply not exist.

We seek a Structural “Signature” that:
- maps hierarchical structure to a point in some low dimensional space.
- captures local structure to support indexing in the presence of occlusion.
- is unique, i.e., different structures have different signatures.
- is efficiently computed.
- is stable, i.e., small structural perturbations due to noise result in small perturbations of the signature; moreover, perturbations at coarser levels have more impact than those at finer levels.

Naïve Characterizations of Structure

<table>
<thead>
<tr>
<th>Measure</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum degree</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Maximum degree</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Average degree</td>
<td>1.87</td>
<td>1.87</td>
</tr>
<tr>
<td>Average degree internal</td>
<td>2.86</td>
<td>2.86</td>
</tr>
<tr>
<td>Degree variance</td>
<td>0.82</td>
<td>0.82</td>
</tr>
</tbody>
</table>

The Eigenspace of a Graph

**Preliminaries:**
- Let $A_G \in \{0,1\}^{m \times n}$ be the adjacency matrix of the graph $G$ on $m$ vertices.
- Let $H$ be an $n$-vertex graph obtained by adding $n-m$ new vertices and a set of edges to the graph $G$.
- Let $\Psi : \{0,1\}^{m \times n} \rightarrow \{0,1\}^{n \times m}$ be a lifting operator which transforms a subspace of $\mathbb{R}^{m \times n}$ to a subspace of $\mathbb{R}^{n \times m}$.

We call this operator spectrum preserving if the non-zero eigenvalues of any matrix $A \in \{0,1\}^{m \times n}$ and its image with respect to the operator $(\Psi(A))$ are equal.
The Eigenspace of a Graph

Preliminaries:
  • Let $A_G \in \{0,1\}^{m \times n}$ be the adjacency matrix of the graph $G$ on $m$ vertices.
  • Let $H$ be an $n$-vertex graph obtained by adding $n - m$ new vertices and a set of edges to the graph $G$.
  • Let $\Psi: \{0,1\}^{m \times n} \rightarrow \{0,1\}^{n \times n}$ be a lifting operator which transforms a subspace of $\mathbb{R}^{m \times n}$ to a subspace of $\mathbb{R}^{n \times n}$, $n \geq m$.

We call this operator *spectrum preserving* if the non-zero eigenvalues of any matrix $A \in \{0,1\}^{m \times n}$ and its image with respect to the operator $(\Psi(A))$ are equal.

Seeking a Structural “Signature” that:
  • maps hierarchical structure to a point in some low dimensional space.
  • captures local structure to support indexing in the presence of occlusion.
  • is unique, i.e., different structures have different signatures.
  • is efficiently computed.
  • is stable, i.e., small structural perturbations due to noise result in small perturbations of the signature; moreover, perturbations at coarser levels have more impact than those at finer levels.

Any structural change in graph $G$ can be represented in terms of a spectrum preserving operator and a noise matrix:

$$A_H = \Psi(A_G) + E_H$$

where $\Psi(\cdot)$ is a two-step lifting operator:

1. Add $n - m$ zero rows and columns to $A_G$, forming $A'_G$.
2. Compute $A''_G = P A'_G P^T$, for suitable $P$, aligning rows and columns for corresponding vertices in $A_H$ and $\Psi(A_G)$.

The noise matrix, $E_H$, can therefore be represented as:

$$A_H - \Psi(A_G) \in \{0,1\}^{n \times n}$$
Theorem (Wilkinson, 1965):

If $A$ and $A+E$ are $n \times n$ symmetric matrices, then for all $k \in \{1, \ldots, n\}$, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$:

$$
\lambda_k(A) + \lambda_k(E) \leq \lambda_k(A + E) \leq \lambda_k(A) + \lambda_k(E).
$$

For $H$ (perturbed graph) and $G$ (original graph), the above theorem yields (after manipulation):

$$
[\lambda_k(H) - \lambda_k(G)] \leq [\lambda_k(E_H)],
$$

The eigenvalues of a graph are therefore stable under minor perturbations in graph structure.

---

Populating the Model Database (compile time)

Model View $M_1$  Model Database

Back to Our Example

[4.168, 0.0]  [3.414, 3.414]

Construct a Bipartite Graph:

Consider a bipartite graph matching formulation, in which the edges in the query and model graphs are discarded.

Hierarchical structure is seemingly lost, but can be encoded in the edge weights:

$$
W(i, j) = e^{-w_g(d_{geom}(i, j)) + w_t(d_{geom}(i, j))}
$$

Hierarchical structure is seemingly lost, but can be encoded in the edge weights:

$$
W(i, j) = e^{-w_g(d_{geom}(i, j)) + w_t(d_{geom}(i, j))}
$$
Algorithm Overview

query and model graphs
bipartite graph
compute maximum cardinality, maximum weight matching
add best edge to solution and...
recursively descend

Illustrative Example

Computed Correspondence
Indexing Experiments

Unoccluded Queries

Occluded Queries

Generic Object Matching

<table>
<thead>
<tr>
<th>Index</th>
<th>Unindex</th>
<th>Unindex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.23</td>
<td>2.93</td>
<td>4.23</td>
</tr>
<tr>
<td>3.32</td>
<td>5.23</td>
<td>6.32</td>
</tr>
<tr>
<td>2.32</td>
<td>4.23</td>
<td>5.32</td>
</tr>
<tr>
<td>3.32</td>
<td>4.23</td>
<td>5.32</td>
</tr>
<tr>
<td>4.32</td>
<td>5.32</td>
<td>6.32</td>
</tr>
<tr>
<td>5.32</td>
<td>6.32</td>
<td>7.32</td>
</tr>
<tr>
<td>6.32</td>
<td>7.32</td>
<td>8.32</td>
</tr>
<tr>
<td>7.32</td>
<td>8.32</td>
<td>9.32</td>
</tr>
<tr>
<td>8.32</td>
<td>9.32</td>
<td>10.32</td>
</tr>
<tr>
<td>9.32</td>
<td>10.32</td>
<td>11.32</td>
</tr>
<tr>
<td>10.32</td>
<td>11.32</td>
<td>12.32</td>
</tr>
<tr>
<td>11.32</td>
<td>12.32</td>
<td>13.32</td>
</tr>
<tr>
<td>12.32</td>
<td>13.32</td>
<td>14.32</td>
</tr>
<tr>
<td>13.32</td>
<td>14.32</td>
<td>15.32</td>
</tr>
<tr>
<td>14.32</td>
<td>15.32</td>
<td>16.32</td>
</tr>
<tr>
<td>15.32</td>
<td>16.32</td>
<td>17.32</td>
</tr>
<tr>
<td>16.32</td>
<td>17.32</td>
<td>18.32</td>
</tr>
<tr>
<td>17.32</td>
<td>18.32</td>
<td>19.32</td>
</tr>
<tr>
<td>18.32</td>
<td>19.32</td>
<td>20.32</td>
</tr>
<tr>
<td>19.32</td>
<td>20.32</td>
<td>21.32</td>
</tr>
<tr>
<td>20.32</td>
<td>21.32</td>
<td>22.32</td>
</tr>
<tr>
<td>21.32</td>
<td>22.32</td>
<td>23.32</td>
</tr>
<tr>
<td>22.32</td>
<td>23.32</td>
<td>24.32</td>
</tr>
<tr>
<td>23.32</td>
<td>24.32</td>
<td>25.32</td>
</tr>
<tr>
<td>24.32</td>
<td>25.32</td>
<td>26.32</td>
</tr>
<tr>
<td>25.32</td>
<td>26.32</td>
<td>27.32</td>
</tr>
<tr>
<td>26.32</td>
<td>27.32</td>
<td>28.32</td>
</tr>
<tr>
<td>27.32</td>
<td>28.32</td>
<td>29.32</td>
</tr>
<tr>
<td>28.32</td>
<td>29.32</td>
<td>30.32</td>
</tr>
<tr>
<td>29.32</td>
<td>30.32</td>
<td>31.32</td>
</tr>
<tr>
<td>30.32</td>
<td>31.32</td>
<td>32.32</td>
</tr>
<tr>
<td>31.32</td>
<td>32.32</td>
<td>33.32</td>
</tr>
<tr>
<td>32.32</td>
<td>33.32</td>
<td>34.32</td>
</tr>
<tr>
<td>33.32</td>
<td>34.32</td>
<td>35.32</td>
</tr>
<tr>
<td>34.32</td>
<td>35.32</td>
<td>36.32</td>
</tr>
<tr>
<td>35.32</td>
<td>36.32</td>
<td>37.32</td>
</tr>
<tr>
<td>36.32</td>
<td>37.32</td>
<td>38.32</td>
</tr>
<tr>
<td>37.32</td>
<td>38.32</td>
<td>39.32</td>
</tr>
<tr>
<td>38.32</td>
<td>39.32</td>
<td>40.32</td>
</tr>
<tr>
<td>39.32</td>
<td>40.32</td>
<td>41.32</td>
</tr>
<tr>
<td>40.32</td>
<td>41.32</td>
<td>42.32</td>
</tr>
<tr>
<td>41.32</td>
<td>42.32</td>
<td>43.32</td>
</tr>
<tr>
<td>42.32</td>
<td>43.32</td>
<td>44.32</td>
</tr>
<tr>
<td>43.32</td>
<td>44.32</td>
<td>45.32</td>
</tr>
<tr>
<td>44.32</td>
<td>45.32</td>
<td>46.32</td>
</tr>
<tr>
<td>45.32</td>
<td>46.32</td>
<td>47.32</td>
</tr>
<tr>
<td>46.32</td>
<td>47.32</td>
<td>48.32</td>
</tr>
<tr>
<td>47.32</td>
<td>48.32</td>
<td>49.32</td>
</tr>
<tr>
<td>48.32</td>
<td>49.32</td>
<td>50.32</td>
</tr>
<tr>
<td>49.32</td>
<td>50.32</td>
<td>51.32</td>
</tr>
<tr>
<td>50.32</td>
<td>51.32</td>
<td>52.32</td>
</tr>
<tr>
<td>51.32</td>
<td>52.32</td>
<td>53.32</td>
</tr>
<tr>
<td>52.32</td>
<td>53.32</td>
<td>54.32</td>
</tr>
<tr>
<td>53.32</td>
<td>54.32</td>
<td>55.32</td>
</tr>
<tr>
<td>54.32</td>
<td>55.32</td>
<td>56.32</td>
</tr>
<tr>
<td>55.32</td>
<td>56.32</td>
<td>57.32</td>
</tr>
<tr>
<td>56.32</td>
<td>57.32</td>
<td>58.32</td>
</tr>
<tr>
<td>57.32</td>
<td>58.32</td>
<td>59.32</td>
</tr>
<tr>
<td>58.32</td>
<td>59.32</td>
<td>60.32</td>
</tr>
<tr>
<td>59.32</td>
<td>60.32</td>
<td>61.32</td>
</tr>
<tr>
<td>60.32</td>
<td>61.32</td>
<td>62.32</td>
</tr>
<tr>
<td>61.32</td>
<td>62.32</td>
<td>63.32</td>
</tr>
<tr>
<td>62.32</td>
<td>63.32</td>
<td>64.32</td>
</tr>
<tr>
<td>63.32</td>
<td>64.32</td>
<td>65.32</td>
</tr>
<tr>
<td>64.32</td>
<td>65.32</td>
<td>66.32</td>
</tr>
<tr>
<td>65.32</td>
<td>66.32</td>
<td>67.32</td>
</tr>
<tr>
<td>66.32</td>
<td>67.32</td>
<td>68.32</td>
</tr>
<tr>
<td>67.32</td>
<td>68.32</td>
<td>69.32</td>
</tr>
<tr>
<td>68.32</td>
<td>69.32</td>
<td>70.32</td>
</tr>
<tr>
<td>69.32</td>
<td>70.32</td>
<td>71.32</td>
</tr>
<tr>
<td>70.32</td>
<td>71.32</td>
<td>72.32</td>
</tr>
<tr>
<td>71.32</td>
<td>72.32</td>
<td>73.32</td>
</tr>
<tr>
<td>72.32</td>
<td>73.32</td>
<td>74.32</td>
</tr>
<tr>
<td>73.32</td>
<td>74.32</td>
<td>75.32</td>
</tr>
<tr>
<td>74.32</td>
<td>75.32</td>
<td>76.32</td>
</tr>
<tr>
<td>75.32</td>
<td>76.32</td>
<td>77.32</td>
</tr>
<tr>
<td>76.32</td>
<td>77.32</td>
<td>78.32</td>
</tr>
<tr>
<td>77.32</td>
<td>78.32</td>
<td>79.32</td>
</tr>
<tr>
<td>78.32</td>
<td>79.32</td>
<td>80.32</td>
</tr>
<tr>
<td>79.32</td>
<td>80.32</td>
<td>81.32</td>
</tr>
<tr>
<td>80.32</td>
<td>81.32</td>
<td>82.32</td>
</tr>
<tr>
<td>81.32</td>
<td>82.32</td>
<td>83.32</td>
</tr>
<tr>
<td>82.32</td>
<td>83.32</td>
<td>84.32</td>
</tr>
<tr>
<td>83.32</td>
<td>84.32</td>
<td>85.32</td>
</tr>
<tr>
<td>84.32</td>
<td>85.32</td>
<td>86.32</td>
</tr>
<tr>
<td>85.32</td>
<td>86.32</td>
<td>87.32</td>
</tr>
<tr>
<td>86.32</td>
<td>87.32</td>
<td>88.32</td>
</tr>
<tr>
<td>87.32</td>
<td>88.32</td>
<td>89.32</td>
</tr>
<tr>
<td>88.32</td>
<td>89.32</td>
<td>90.32</td>
</tr>
<tr>
<td>89.32</td>
<td>90.32</td>
<td>91.32</td>
</tr>
<tr>
<td>90.32</td>
<td>91.32</td>
<td>92.32</td>
</tr>
<tr>
<td>91.32</td>
<td>92.32</td>
<td>93.32</td>
</tr>
<tr>
<td>92.32</td>
<td>93.32</td>
<td>94.32</td>
</tr>
<tr>
<td>93.32</td>
<td>94.32</td>
<td>95.32</td>
</tr>
<tr>
<td>94.32</td>
<td>95.32</td>
<td>96.32</td>
</tr>
<tr>
<td>95.32</td>
<td>96.32</td>
<td>97.32</td>
</tr>
<tr>
<td>96.32</td>
<td>97.32</td>
<td>98.32</td>
</tr>
<tr>
<td>97.32</td>
<td>98.32</td>
<td>99.32</td>
</tr>
<tr>
<td>98.32</td>
<td>99.32</td>
<td>100.32</td>
</tr>
</tbody>
</table>
Example Correspondences

Conclusions

• The eigenvalue characterization of hierarchical graph structure yields a powerful structural indexing mechanism applicable to the recognition of hierarchical image structures in computer vision.

• This same eigenvalue characterization forms the heart of our matching algorithm that can determine node correspondence in the presence of clutter, occlusion, and spurious noise.

• This framework has been successfully applied to multiple generic object recognition domains.

Summary of the Approaches

Combinatorial Methods

Bipartite Matching:

• Discards graph structure, setting up a pure correspondence problem for which efficient max cardinality, max weight matching algorithms are available.

• To enforce structural and hierarchical constraints, bipartite matching can be structured coarse-to-fine, or structural information can be absorbed into the nodes.

• Occlusion can be handled, but limited indexing support provided, and limited invariance to noise.

Tree Edit Distance:

• Reformulates ordered tree matching as a string edit-distance matching problem, for which efficient solutions are known.

• Editing out clutter can slow down algorithm, adding a large edit distance when the embedded object may be identical to the model.

• Robust to noise and minor occlusion, but no indexing support.
Optimization Methods

**Graduated Assignment:**
- efficient continuous optimization formulation that handles node and edge weights.
- exploits powerful doubly stochastic matrix convergence result by Sinkhorn.
- can handle lost nodes and spurious edges between nodes, but cannot accommodate spurious nodes within target structure.
- no indexing support.

**Maximum Clique:**
- handles rooted (hierarchical) trees, free trees, attributed trees.
- can handle arbitrary occlusion, but not noise in target structure.
- exploits powerful Motzkin-Straus mapping between discrete and continuous problems.
- no indexing support.

Algebraic Methods

**Spectral Methods:**
- low-dimensional characterization of graph structure.
- stable to minor structural perturbation.
- strong indexing support
- limited many-to-many matching support.

Choosing the Right Framework

Questions you should be asking yourself:
- should I even use a graph to model my features?
- exact or inexact matching?
- extent and type of noise?
- extent of occlusion?
- hierarchical or flat?
- trees or DAG’s?
- computational complexity?
- indexing support needed? (target detection or db retrieval)
- probabilistic components?
- topological or geometric matching?
- categorical or exemplar-based recognition?
- edge weights? node labels?
Open Problems

Graph indexing is rarely addressed by the graph algorithms community or the vision community.

Many-to-many matching is the real vision problem. We need matching frameworks that can support it, but more importantly, we need feature abstraction rules to group nodes.

Systematic evaluation of recognition performance (indexing efficiency, matching correctness) w.r.t. noise, occlusion, database scaling, is rare. The vision community needs benchmark graphs that reflect real vision problems.