Review of Goals

• Mine the photo tag relationship in Facebook for clusters.
• Use these clusters with additional information in comments, album titles, and photo captions to provide a new way of exploring photos and relationships.
Why Photos?

• Unlike general Friend relationship, photos capture a significant amount of context (a picture is worth a thousand words).
  ▫ Typically implies shared presence in space/time.
  ▫ Could be associated with a place or an event.
  ▫ Repeated occurrences can be attributed to a group membership.
There are exceptions...
Original Plan

- Get Facebook API key
- Get Google AppEngine Account
- ...
- Profit

![Diagram of Photos, People, Albums, Comments connected with arrows]
Well...

- The latency from Facebook troublesome.
  - They do fail after 45 seconds...
- Google caps at 30 seconds.
  - Yay! Dreamhost (profit -= ...)
- Facebook silent limits.
- We’re not Web 2.0 AJAX ninjas...
Expanded Model

![Diagram showing relationships between Cooccurrence, RefreshState, Person, Albums, TagOccurrence, and Pictures with numbers indicating connections and counts.]
Extended pyfacebook for multiquery

- `queries['tags'] = 'SELECT pid, subject, created, text FROM photo_tag WHERE subject IN (%s);' % uid_str`
- `queries['aids'] = 'SELECT aid, owner, name, description, created, size FROM album WHERE owner IN (%s);' % uid_str`
- `queries['pids'] = 'SELECT pid, aid, owner, link FROM photo WHERE aid IN (%s);' % aid_str`
- `queries['more_tags'] = 'SELECT pid, subject, created, text FROM photo_tag WHERE pid IN (SELECT pid FROM #pids);'`
- `queries['more_names'] = 'SELECT uid, name FROM user WHERE uid IN (SELECT subject FROM #more_tags);'`
DB’s can be nice...

```
SELECT
  ta.subject, ta.text, tb.subject, tb.text, COUNT(*)
FROM
  friendview_tagoccurrence ta JOIN (friendview_tagoccurrence tb)
    ON (ta.pid = tb.pid AND
        ta.subject != tb.subject AND
        ta.owner = tb.owner)
WHERE
  ta.owner = %s AND ta.subject != '' AND
  tb.subject != ''
GROUP BY ta.subject, tb.subject;
```
Photos by Friend

Rank friends by the number of co-occurrences

Overview of the photos of each friend

Recommend people that appear also with that person
Co-occurences

Summary of the co-occurrences of one's friends

<table>
<thead>
<tr>
<th>Rank of friends per number of co-occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>

[Facebook interface showing co-occurrences]
Friend’s Details

- Link to the profile
- Photos of this friend
- Ranked list of people that co-occurs with this friend
Photo Details

Link to the Facebook's photo page

All the people that co-occurs in this photo

See this photo in Facebook
People in this photo:
Current Solutions
MyLifePix Analysis
Gephi
Future Work

- Implement clustering algorithms
- Show the graph
- Migrate back to Google App Engine
- Improve the performance
Thanks!

Questions?