Overview of this week

• Debugging tips for ML algorithms
• Graph algorithms
  – A prototypical graph algorithm: PageRank
    • In memory
    • Putting more and more on disk ...
  – Sampling from a graph
    • What is a good sample? (graph statistics)
    • What methods work? (PPR/RWR)
    • HW: PageRank-Nibble method + Gephi
Common statistics for graphs

William Cohen
Why I’m talking about graphs

• Lots of large data is graphs
  – Facebook, Twitter, citation data, and other social networks
  – The web, the blogosphere, the semantic web, Freebase, Wikipedia, Twitter, and other information networks
  – Text corpora (like RCV1), large datasets with discrete feature values, and other bipartite networks
    • nodes = documents or words
    • links connect document \(\rightarrow\) word or word \(\rightarrow\) document
  – Computer networks, biological networks (proteins, ecosystems, brains, ...), ...
  – Heterogeneous networks with multiple types of nodes
    • people, groups, documents
A question

• How do you explore a dataset?
  – compute statistics (e.g., feature histograms, conditional feature histograms, correlation coefficients, ...)
  – sample and inspect
    • run a bunch of small-scale experiments

• How do you explore a graph?
  – compute statistics (degree distribution, ...)
  – sample and inspect
    • how do you sample?
A question

• How do you explore a graph?
  – compute statistics (degree distribution, ...)
  – sample and inspect
    • how do you sample?
• This is the next assignment:
  – Sampling from a graph
<table>
<thead>
<tr>
<th>network</th>
<th>type</th>
<th>$n$</th>
<th>$m$</th>
<th>$z$</th>
<th>$\ell$</th>
<th>$\alpha$</th>
<th>$C^{(1)}$</th>
<th>$C^{(2)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>film actors</td>
<td>undirected</td>
<td>449 913</td>
<td>25 516 482</td>
<td>113.43</td>
<td>3.48</td>
<td>2.3</td>
<td>0.20</td>
<td>0.78</td>
</tr>
<tr>
<td>company directors</td>
<td>undirected</td>
<td>7 673</td>
<td>55 392</td>
<td>14.44</td>
<td>4.60</td>
<td>–</td>
<td>0.59</td>
<td>0.88</td>
</tr>
<tr>
<td>math coauthorship</td>
<td>undirected</td>
<td>253 339</td>
<td>496 489</td>
<td>3.92</td>
<td>7.57</td>
<td>–</td>
<td>0.15</td>
<td>0.34</td>
</tr>
<tr>
<td>physics coauthorship</td>
<td>undirected</td>
<td>52 909</td>
<td>245 300</td>
<td>9.27</td>
<td>6.19</td>
<td>–</td>
<td>0.45</td>
<td>0.56</td>
</tr>
<tr>
<td>biology coauthorship</td>
<td>undirected</td>
<td>1 520 251</td>
<td>11 803 064</td>
<td>15.53</td>
<td>4.92</td>
<td>–</td>
<td>0.088</td>
<td>0.60</td>
</tr>
<tr>
<td>telephone call graph</td>
<td>undirected</td>
<td>47 000 000</td>
<td>80 000 000</td>
<td>3.16</td>
<td>-</td>
<td>2.1</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>email messages</td>
<td>directed</td>
<td>59 912</td>
<td>86 300</td>
<td>1.44</td>
<td>4.95</td>
<td>1.5/2.0</td>
<td>0.16</td>
<td></td>
</tr>
<tr>
<td>email address books</td>
<td>directed</td>
<td>16 881</td>
<td>57 029</td>
<td>3.38</td>
<td>5.22</td>
<td>–</td>
<td>0.17</td>
<td>0.13</td>
</tr>
<tr>
<td>student relationships</td>
<td>undirected</td>
<td>573</td>
<td>477</td>
<td>1.66</td>
<td>16.01</td>
<td>–</td>
<td>0.005</td>
<td>0.001</td>
</tr>
<tr>
<td>sexual contacts</td>
<td>undirected</td>
<td>2 810</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WWW nd.edu</td>
<td>directed</td>
<td>269 504</td>
<td>1 497 135</td>
<td>5.55</td>
<td>11.27</td>
<td>2.1/2.4</td>
<td>0.11</td>
<td>0.29</td>
</tr>
<tr>
<td>WWW Altavista</td>
<td>directed</td>
<td>203 549 046</td>
<td>2 130 000 000</td>
<td>10.46</td>
<td>16.18</td>
<td>2.1/2.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>citation network</td>
<td>directed</td>
<td>783 339</td>
<td>6 716 198</td>
<td>8.57</td>
<td>-</td>
<td>3.0/–</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roget’s Thesaurus</td>
<td>directed</td>
<td>1 022</td>
<td>5 103</td>
<td>4.99</td>
<td>4.87</td>
<td>–</td>
<td>0.13</td>
<td>0.15</td>
</tr>
<tr>
<td>word co-occurrence</td>
<td>undirected</td>
<td>460 902</td>
<td>17 000 000</td>
<td>70.13</td>
<td>-</td>
<td>2.7</td>
<td>0.44</td>
<td></td>
</tr>
<tr>
<td>Internet</td>
<td>undirected</td>
<td>10 697</td>
<td>31 992</td>
<td>5.98</td>
<td>3.31</td>
<td>2.5</td>
<td>0.035</td>
<td>0.39</td>
</tr>
<tr>
<td>power grid</td>
<td>undirected</td>
<td>4 941</td>
<td>6 594</td>
<td>2.67</td>
<td>18.99</td>
<td>–</td>
<td>0.10</td>
<td>0.080</td>
</tr>
<tr>
<td>train routes</td>
<td>undirected</td>
<td>587</td>
<td>19 603</td>
<td>66.79</td>
<td>2.16</td>
<td>–</td>
<td></td>
<td>0.69</td>
</tr>
<tr>
<td>software packages</td>
<td>directed</td>
<td>1 439</td>
<td>1 723</td>
<td>1.20</td>
<td>2.42</td>
<td>1.6/1.4</td>
<td>0.070</td>
<td>0.082</td>
</tr>
<tr>
<td>software classes</td>
<td>directed</td>
<td>1 377</td>
<td>2 213</td>
<td>1.61</td>
<td>1.51</td>
<td>–</td>
<td>0.033</td>
<td>0.012</td>
</tr>
<tr>
<td>electronic circuits</td>
<td>undirected</td>
<td>24 097</td>
<td>53 248</td>
<td>4.34</td>
<td>11.05</td>
<td>3.0</td>
<td>0.010</td>
<td>0.030</td>
</tr>
<tr>
<td>peer-to-peer network</td>
<td>undirected</td>
<td>880</td>
<td>1 296</td>
<td>1.47</td>
<td>4.28</td>
<td>2.1</td>
<td>0.012</td>
<td>0.011</td>
</tr>
<tr>
<td>metabolic network</td>
<td>undirected</td>
<td>765</td>
<td>3 686</td>
<td>9.64</td>
<td>2.56</td>
<td>2.2</td>
<td>0.090</td>
<td>0.67</td>
</tr>
<tr>
<td>protein interactions</td>
<td>undirected</td>
<td>2 115</td>
<td>2 240</td>
<td>2.12</td>
<td>6.80</td>
<td>2.4</td>
<td>0.072</td>
<td>0.071</td>
</tr>
<tr>
<td>marine food web</td>
<td>directed</td>
<td>135</td>
<td>598</td>
<td>4.43</td>
<td>2.05</td>
<td>–</td>
<td>0.16</td>
<td>0.23</td>
</tr>
<tr>
<td>freshwater food web</td>
<td>directed</td>
<td>92</td>
<td>997</td>
<td>10.84</td>
<td>1.90</td>
<td>–</td>
<td>0.20</td>
<td>0.087</td>
</tr>
<tr>
<td>neural network</td>
<td>directed</td>
<td>307</td>
<td>2 359</td>
<td>7.68</td>
<td>3.97</td>
<td>–</td>
<td>0.18</td>
<td>0.28</td>
</tr>
</tbody>
</table>
Degree distribution

- Plot cumulative degree
  - X axis is degree
  - Y axis is #nodes that have degree at least $k$
- Typically use a log-log scale
  - Straight lines are a power law; normal curve dives to zero at some point
  - Left: trust network in epinions web site from Richardson & Domingos
FIG. 6 Cumulative degree distributions for six different networks. The horizontal axis for each panel is vertex degree $k$ (or in-degree for the citation and Web networks, which are directed) and the vertical axis is the cumulative probability distribution of degrees, i.e., the fraction of vertices that have degree greater than or equal to $k$. The networks shown are: (a) the collaboration network of mathematicians [182]; (b) citations between 1981 and 1997 to all papers cataloged by the Institute for Scientific Information [351]; (c) a 300 million vertex subset of the World Wide Web, circa 1999 [74]; (d) the Internet at the level of autonomous systems, April 1999 [86]; (e) the power grid of the western United States [416]; (f) the interaction network of proteins in the metabolism of the yeast *S. Cerevisiae* [212]. Of these networks, three of them, (c), (d) and (f), appear to have power-law degree distributions, as indicated by their approximately straight-line forms on the doubly logarithmic scales, and one (b) has a power-law tail but deviates markedly from power-law behavior for small degree. Network (e) has an exponential degree distribution (note the log-linear scales used in this panel) and network (a) appears to have a truncated power-law degree distribution of some type, or possibly two separate power-law regimes with different exponents.
<table>
<thead>
<tr>
<th>network</th>
<th>type</th>
<th>$n$</th>
<th>$m$</th>
<th>$z$</th>
<th>$\ell$</th>
<th>$\alpha$</th>
<th>$C^{(1)}$</th>
<th>$C^{(2)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>film actors</td>
<td>undirected</td>
<td>449,913</td>
<td>25,516,482</td>
<td>113.43</td>
<td>3.48</td>
<td>2.3</td>
<td>0.20</td>
<td>0.78</td>
</tr>
<tr>
<td>company directors</td>
<td>undirected</td>
<td>7,673</td>
<td>55,392</td>
<td>14.44</td>
<td>4.60</td>
<td>–</td>
<td>0.59</td>
<td>0.88</td>
</tr>
<tr>
<td>math coauthorship</td>
<td>undirected</td>
<td>253,339</td>
<td>496,489</td>
<td>3.92</td>
<td>7.57</td>
<td>–</td>
<td>0.15</td>
<td>0.34</td>
</tr>
<tr>
<td>physics coauthorship</td>
<td>undirected</td>
<td>52,909</td>
<td>245,300</td>
<td>9.27</td>
<td>6.19</td>
<td>–</td>
<td>0.45</td>
<td>0.56</td>
</tr>
<tr>
<td>biology coauthorship</td>
<td>undirected</td>
<td>1,520,251</td>
<td>11,803,064</td>
<td>15.53</td>
<td>4.92</td>
<td>–</td>
<td>0.088</td>
<td>0.60</td>
</tr>
<tr>
<td>telephone call graph</td>
<td>undirected</td>
<td>47,000,000</td>
<td>80,000,000</td>
<td>3.16</td>
<td>2.1</td>
<td>–</td>
<td>0.16</td>
<td>0.16</td>
</tr>
<tr>
<td>email messages</td>
<td>directed</td>
<td>59,912</td>
<td>86,300</td>
<td>1.44</td>
<td>4.95</td>
<td>1.5/2.0</td>
<td>0.17</td>
<td>0.13</td>
</tr>
<tr>
<td>email address books</td>
<td>directed</td>
<td>16,881</td>
<td>57,029</td>
<td>3.38</td>
<td>5.22</td>
<td>–</td>
<td>0.17</td>
<td>0.13</td>
</tr>
<tr>
<td>student relationships</td>
<td>undirected</td>
<td>573</td>
<td>477</td>
<td>1.66</td>
<td>16.01</td>
<td>–</td>
<td>0.005</td>
<td>0.001</td>
</tr>
<tr>
<td>sexual contacts</td>
<td>undirected</td>
<td>2,810</td>
<td></td>
<td></td>
<td></td>
<td>3.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WWW nd.edu</td>
<td>directed</td>
<td>269,504</td>
<td>1,497,135</td>
<td>5.55</td>
<td>11.27</td>
<td>2.1/2.4</td>
<td>0.11</td>
<td>0.29</td>
</tr>
<tr>
<td>WWW Altavista</td>
<td>directed</td>
<td>203,549,046</td>
<td>2,130,000,000</td>
<td>10.46</td>
<td>16.18</td>
<td>2.1/2.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>citation network</td>
<td>directed</td>
<td>783,339</td>
<td>6,716,198</td>
<td>8.57</td>
<td></td>
<td>3.0/–</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roget’s Thesaurus</td>
<td>directed</td>
<td>1,022</td>
<td>5,103</td>
<td>4.99</td>
<td>4.87</td>
<td>–</td>
<td>0.13</td>
<td>0.15</td>
</tr>
<tr>
<td>word co-occurrence</td>
<td>undirected</td>
<td>460,902</td>
<td>17,000,000</td>
<td>70.13</td>
<td></td>
<td>2.7</td>
<td></td>
<td>0.44</td>
</tr>
<tr>
<td>Internet</td>
<td>undirected</td>
<td>10,697</td>
<td>31,992</td>
<td>5.98</td>
<td>3.31</td>
<td>2.5</td>
<td>0.035</td>
<td>0.39</td>
</tr>
<tr>
<td>power grid</td>
<td>undirected</td>
<td>4,941</td>
<td>6,594</td>
<td>2.67</td>
<td>18.99</td>
<td>–</td>
<td>0.10</td>
<td>0.080</td>
</tr>
<tr>
<td>train routes</td>
<td>undirected</td>
<td>587</td>
<td>19,603</td>
<td>66.79</td>
<td>2.16</td>
<td>–</td>
<td>0.69</td>
<td></td>
</tr>
<tr>
<td>software packages</td>
<td>directed</td>
<td>1,439</td>
<td>1,723</td>
<td>1.20</td>
<td>2.42</td>
<td>1.6/1.4</td>
<td>0.070</td>
<td>0.082</td>
</tr>
<tr>
<td>software classes</td>
<td>directed</td>
<td>1,377</td>
<td>2,213</td>
<td>1.61</td>
<td>1.51</td>
<td>–</td>
<td>0.033</td>
<td>0.012</td>
</tr>
<tr>
<td>electronic circuits</td>
<td>undirected</td>
<td>24,097</td>
<td>53,248</td>
<td>4.34</td>
<td>11.05</td>
<td>3.0</td>
<td>0.010</td>
<td>0.030</td>
</tr>
<tr>
<td>peer-to-peer network</td>
<td>undirected</td>
<td>880</td>
<td>1,296</td>
<td>1.47</td>
<td>4.28</td>
<td>2.1</td>
<td>0.012</td>
<td>0.011</td>
</tr>
<tr>
<td>metabolic network</td>
<td>undirected</td>
<td>765</td>
<td>3,686</td>
<td>9.64</td>
<td>2.56</td>
<td>2.2</td>
<td>0.090</td>
<td>0.67</td>
</tr>
<tr>
<td>protein interactions</td>
<td>undirected</td>
<td>2,115</td>
<td>2,240</td>
<td>2.12</td>
<td>6.80</td>
<td>2.4</td>
<td>0.072</td>
<td>0.071</td>
</tr>
<tr>
<td>marine food web</td>
<td>directed</td>
<td>135</td>
<td>598</td>
<td>4.43</td>
<td>2.05</td>
<td>–</td>
<td>0.16</td>
<td>0.23</td>
</tr>
<tr>
<td>freshwater food web</td>
<td>directed</td>
<td>92</td>
<td>997</td>
<td>10.84</td>
<td>1.90</td>
<td>–</td>
<td>0.20</td>
<td>0.087</td>
</tr>
<tr>
<td>neural network</td>
<td>directed</td>
<td>307</td>
<td>2,359</td>
<td>7.68</td>
<td>3.97</td>
<td>–</td>
<td>0.18</td>
<td>0.28</td>
</tr>
</tbody>
</table>
Homophily

• Another def’n: excess edges between common neighbors of $v$

$CC(v) = \frac{\text{# triangles connected to } v}{\text{# pairs connected to } v}$

$CC(V, E) = \frac{1}{|V|} \sum_{v} CC(v)$

$CC'(V, E) = \frac{\text{# triangles in graph}}{\text{# length 3 paths in graph}}$
<table>
<thead>
<tr>
<th>network</th>
<th>type</th>
<th>$n$</th>
<th>$m$</th>
<th>$z$</th>
<th>$\ell$</th>
<th>$\alpha$</th>
<th>$C^{(1)}$</th>
<th>$C^{(2)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>film actors</td>
<td>undirected</td>
<td>449,913</td>
<td>25,516,482</td>
<td>113.43</td>
<td>3.48</td>
<td>2.3</td>
<td>0.20</td>
<td>0.78</td>
</tr>
<tr>
<td>company directors</td>
<td>undirected</td>
<td>7,673</td>
<td>55,392</td>
<td>14.44</td>
<td>4.60</td>
<td>-</td>
<td>0.59</td>
<td>0.88</td>
</tr>
<tr>
<td>math coauthorship</td>
<td>undirected</td>
<td>253,339</td>
<td>496,489</td>
<td>3.92</td>
<td>7.57</td>
<td>-</td>
<td>0.15</td>
<td>0.34</td>
</tr>
<tr>
<td>physics coauthorship</td>
<td>undirected</td>
<td>52,909</td>
<td>245,300</td>
<td>9.27</td>
<td>6.19</td>
<td>-</td>
<td>0.45</td>
<td>0.56</td>
</tr>
<tr>
<td>biology coauthorship</td>
<td>undirected</td>
<td>1,520,251</td>
<td>11,803,064</td>
<td>15.53</td>
<td>4.92</td>
<td>-</td>
<td>0.088</td>
<td>0.60</td>
</tr>
<tr>
<td>telephone call graph</td>
<td>undirected</td>
<td>47,000,000</td>
<td>80,000,000</td>
<td>3.16</td>
<td>2.1</td>
<td>-</td>
<td>0.16</td>
<td></td>
</tr>
<tr>
<td>email messages</td>
<td>directed</td>
<td>59,912</td>
<td>86,300</td>
<td>1.44</td>
<td>4.95</td>
<td>1.5/2.0</td>
<td>0.17</td>
<td>0.13</td>
</tr>
<tr>
<td>email address books</td>
<td>directed</td>
<td>16,881</td>
<td>57,029</td>
<td>3.38</td>
<td>5.22</td>
<td>-</td>
<td>0.005</td>
<td>0.001</td>
</tr>
<tr>
<td>student relationships</td>
<td>undirected</td>
<td>573</td>
<td>477</td>
<td>1.66</td>
<td>16.01</td>
<td>-</td>
<td>3.2</td>
<td></td>
</tr>
<tr>
<td>sexual contacts</td>
<td>undirected</td>
<td>2,810</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WWW nd.edu</td>
<td>directed</td>
<td>269,504</td>
<td>1,497,135</td>
<td>5.55</td>
<td>11.27</td>
<td>2.1/2.4</td>
<td>0.11</td>
<td>0.29</td>
</tr>
<tr>
<td>WWW Altavista</td>
<td>directed</td>
<td>2,035,490,046</td>
<td>2,130,000,000</td>
<td>10.46</td>
<td>16.18</td>
<td>2.1/2.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>citation network</td>
<td>directed</td>
<td>783,339</td>
<td>6,716,198</td>
<td>8.57</td>
<td></td>
<td>3.0/-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roget’s Thesaurus</td>
<td>directed</td>
<td>1,022</td>
<td>5,103</td>
<td>4.99</td>
<td>4.87</td>
<td>-</td>
<td>0.13</td>
<td>0.15</td>
</tr>
<tr>
<td>word co-occurrence</td>
<td>undirected</td>
<td>460,902</td>
<td>17,000,000</td>
<td>70.13</td>
<td></td>
<td>2.7</td>
<td>0.035</td>
<td>0.39</td>
</tr>
<tr>
<td>Internet</td>
<td>undirected</td>
<td>10,697</td>
<td>31,992</td>
<td>5.98</td>
<td>3.31</td>
<td>2.5</td>
<td>0.035</td>
<td>0.39</td>
</tr>
<tr>
<td>power grid</td>
<td>undirected</td>
<td>4,941</td>
<td>6,594</td>
<td>2.67</td>
<td>18.99</td>
<td>-</td>
<td>0.10</td>
<td>0.080</td>
</tr>
<tr>
<td>train routes</td>
<td>undirected</td>
<td>587</td>
<td>19,603</td>
<td>66.79</td>
<td>2.16</td>
<td>-</td>
<td>0.070</td>
<td>0.082</td>
</tr>
<tr>
<td>software packages</td>
<td>directed</td>
<td>1,439</td>
<td>1,723</td>
<td>1.20</td>
<td>2.42</td>
<td>1.6/1.4</td>
<td>0.033</td>
<td>0.012</td>
</tr>
<tr>
<td>software classes</td>
<td>directed</td>
<td>1,377</td>
<td>2,213</td>
<td>1.61</td>
<td>1.51</td>
<td>-</td>
<td>0.010</td>
<td>0.030</td>
</tr>
<tr>
<td>electronic circuits</td>
<td>undirected</td>
<td>24,097</td>
<td>53,248</td>
<td>4.34</td>
<td>11.05</td>
<td>3.0</td>
<td>0.012</td>
<td>0.011</td>
</tr>
<tr>
<td>peer-to-peer network</td>
<td>undirected</td>
<td>880</td>
<td>1,296</td>
<td>1.47</td>
<td>4.28</td>
<td>2.1</td>
<td>0.090</td>
<td>0.67</td>
</tr>
<tr>
<td>metabolic network</td>
<td>undirected</td>
<td>765</td>
<td>3,686</td>
<td>9.64</td>
<td>2.56</td>
<td>2.2</td>
<td>0.072</td>
<td>0.071</td>
</tr>
<tr>
<td>protein interactions</td>
<td>undirected</td>
<td>2,115</td>
<td>2,240</td>
<td>2.12</td>
<td>6.80</td>
<td>2.4</td>
<td>0.16</td>
<td>0.23</td>
</tr>
<tr>
<td>marine food web</td>
<td>directed</td>
<td>135</td>
<td>598</td>
<td>4.43</td>
<td>2.05</td>
<td>-</td>
<td>0.20</td>
<td>0.087</td>
</tr>
<tr>
<td>freshwater food web</td>
<td>directed</td>
<td>92</td>
<td>997</td>
<td>10.84</td>
<td>1.90</td>
<td>-</td>
<td>0.18</td>
<td>0.28</td>
</tr>
<tr>
<td>neural network</td>
<td>directed</td>
<td>307</td>
<td>2,359</td>
<td>7.68</td>
<td>3.97</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An important question

• How do you explore a dataset?
  – compute statistics (e.g., feature histograms, conditional feature histograms, correlation coefficients, ...)
  – sample and inspect
    • run a bunch of small-scale experiments

• How do you explore a graph?
  – compute statistics (degree distribution, ...)
  – sample and inspect
    • how do you sample?
Overview of this week

• Debugging tips for ML algorithms
• Graph algorithms
  – A prototypical graph algorithm: PageRank
    • In memory
    • Putting more and more on disk ...
  – Sampling from a graph
    • What is a good sample? (graph statistics)
    • What sampling methods work? (PPR/RWR)
    • HW: PageRank-Nibble method + Gephi
Sampling from Large Graphs

Jure Leskovec
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
jure@cs.cmu.edu

Christos Faloutsos
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
christos@cs.cmu.edu

KDD 2006
Brief summary

• Define *goals* of sampling:
  – “scale-down” – find $G'<G$ with similar statistics
  – “back in time”: for a growing $G$, find $G'<G$ that is similar (statistically) to an earlier version of $G$

• Experiment on real graphs with plausible sampling methods, such as
  – RN – random nodes, sampled uniformly
  – ...

• See how well they perform
Brief summary

• Experiment on real graphs with plausible sampling methods, such as
  – RN – random nodes, sampled uniformly
    • RPN – random nodes, sampled by PageRank
    • RDP – random nodes sampled by in-degree
  – RE – random edges
  – RJ – run PageRank’s “random surfer” for $n$ steps
  – RW – run RWR’s “random surfer” for $n$ steps
  – FF – repeatedly pick $r(i)$ neighbors of $i$ to “burn”, and then recursively sample from them
10% sample – pooled on five datasets
d-statistic measures agreement between distributions

- $D = \max\{|F(x) - F'(x)|\}$ where $F, F'$ are cdf’s
- max over nine different statistics
<table>
<thead>
<tr>
<th></th>
<th>in-deg</th>
<th>out-deg</th>
<th>wcc</th>
<th>scc</th>
<th>hops</th>
<th>sng-val</th>
<th>sng-vec</th>
<th>clust</th>
</tr>
</thead>
<tbody>
<tr>
<td>RN</td>
<td>0.084</td>
<td>0.145</td>
<td>0.814</td>
<td>0.193</td>
<td>0.231</td>
<td>0.079</td>
<td>0.112</td>
<td>0.327</td>
</tr>
<tr>
<td>RPN</td>
<td>0.062</td>
<td>0.097</td>
<td>0.792</td>
<td>0.194</td>
<td>0.200</td>
<td>0.048</td>
<td>0.081</td>
<td>0.243</td>
</tr>
<tr>
<td>RDN</td>
<td>0.110</td>
<td>0.128</td>
<td>0.818</td>
<td>0.193</td>
<td>0.238</td>
<td>0.041</td>
<td>0.048</td>
<td>0.256</td>
</tr>
<tr>
<td>RE</td>
<td>0.216</td>
<td>0.305</td>
<td>0.367</td>
<td>0.206</td>
<td>0.509</td>
<td>0.169</td>
<td>0.192</td>
<td>0.525</td>
</tr>
<tr>
<td>RNE</td>
<td>0.277</td>
<td>0.404</td>
<td>0.390</td>
<td>0.224</td>
<td>0.702</td>
<td>0.255</td>
<td>0.273</td>
<td>0.709</td>
</tr>
<tr>
<td>HYB</td>
<td>0.273</td>
<td>0.394</td>
<td>0.386</td>
<td>0.224</td>
<td>0.683</td>
<td>0.240</td>
<td>0.251</td>
<td>0.670</td>
</tr>
<tr>
<td>RNN</td>
<td>0.179</td>
<td>0.014</td>
<td>0.581</td>
<td>0.206</td>
<td>0.252</td>
<td>0.060</td>
<td>0.255</td>
<td>0.398</td>
</tr>
<tr>
<td>RJ</td>
<td>0.132</td>
<td>0.151</td>
<td>0.771</td>
<td>0.215</td>
<td>0.264</td>
<td>0.076</td>
<td>0.143</td>
<td>0.235</td>
</tr>
<tr>
<td>RW</td>
<td>0.082</td>
<td>0.131</td>
<td>0.685</td>
<td>0.194</td>
<td>0.243</td>
<td>0.049</td>
<td>0.033</td>
<td>0.243</td>
</tr>
<tr>
<td>FF</td>
<td>0.082</td>
<td>0.105</td>
<td>0.664</td>
<td>0.194</td>
<td>0.203</td>
<td>0.038</td>
<td>0.092</td>
<td>0.244</td>
</tr>
</tbody>
</table>
Goal

• An efficient way of running RWR on a large graph
  – can use only “random access”
    • you can ask about the neighbors of a node, you can’t scan thru the graph
    • common situation with APIs
  – leads to a plausible sampling strategy
    • Jure & Christos’s experiments
    • some formal results that justify it....
Local Graph Partitioning using PageRank Vectors

Reid Andersen
University of California, San Diego

Fan Chung
University of California, San Diego

Kevin Lang
Yahoo! Research

FOCS 2006
What is Local Graph Partitioning?

A local graph partitioning algorithm finds a small cut near the given seed(s) with running time depending only on the size of the output.
What is Local Graph Partitioning?

A bidding graph from Yahoo sponsored search

Phrases         Advertiser IDs
  e.g. Margarita Mix  e.g. c8cbfd0bd74ba8cc

On the left are search phrases, on the right are advertisers. Each edge represents a bid by an advertiser on a phrase.

400K phrases, 200K advertisers, and 2 million edges.
What is Local Graph Partitioning?

Submarkets in bidding graph

The bidding graph has submarkets, sets of bidders and phrases that interact mostly with each other.

Phrases about margarita mix        Purveyors of margarita mix

These sets of vertices (containing both advertisers and phrases) have small conductance.
What is Local Graph Partitioning?

Submarkets in the bigging graph
The bidding graph has numerous submarkets, related to real estate, flower delivery, hotels, gambling, ...

It is useful to identify these submarkets.
- Find groups of related phrases to suggest to advertisers.
- Find small submarkets for testing and experimentation.
What is Local Graph Partitioning?

A local graph partitioning algorithm finds a small cut near the given seed(s) with running time depending only on the size of the output.
Key idea: a “sweep”

• Order all vertices in some way $v_{i,1}, v_{i,2}, \ldots$
  – Say, by personalized PageRank from a seed
• Pick a prefix $v_{i,1}, v_{i,2}, \ldots v_{i,k}$ that is “best”
What is a “good” subgraph?

\[ \partial(S) = \{ \{x, y\} \in E \mid x \in S, y \notin S \} \]

the edges leaving S

\[ \Phi(S) = \frac{|\partial(S)|}{\min (\text{vol}(S), 2m - \text{vol}(S))}. \]

- \( \text{vol}(S) \) is sum of \( \text{deg}(x) \) for \( x \) in \( S \)
- for small \( S \): Prob(random edge leaves \( S \))
Key idea: a “sweep”

• Order all vertices in some way \( v_{i,1}, v_{i,2}, \ldots \)
  – Say, by personalized PageRank from a seed

• Pick a prefix \( S=\{ v_{i,1}, v_{i,2}, \ldots v_{i,k} \} \) that is “best”
  – Minimal “conductance” \( \phi(S) \)

You can re-compute conductance incrementally as you add a new vertex so the sweep is fast
Main results of the paper

1. An *approximate* personalized PageRank computation that only touches nodes “near” the seed
   – but has small error relative to the true PageRank vector
2. A proof that a sweep over the approximate PageRank vector finds a cut with conductance $\sqrt{\alpha \ln m}$
   – unless no good cut exists
   • no subset $S$ contains significantly more pass in the approximate PageRank than in a uniform distribution
Result 2 explains Jure & Christos’s experimental results with RW sampling:

- RW approximately picks up a random subcommunity (maybe with some extra nodes)
- Features like clustering coefficient, degree should be representative of the graph as a whole...
- which is roughly a mixture of subcommunities
Main results of the paper

1. An approximate personalized PageRank computation that only touches nodes “near” the seed
   – but has small error relative to the true PageRank vector

This is a very useful technique to know about...
Random Walks

\( G : \) a graph

\( P : \) transition probability matrix

\[
P(u, v) = \begin{cases} 
  \frac{1}{d_u} & \text{if } u \to v, \\
  0 & \text{otherwise.}
\end{cases}
\]

\( d_u \) := the degree of \( u \).

A lazy walk:

\[
W = \frac{I + P}{2}
\]

avoids messy “dead ends”….
Random Walks: PageRank

A (bored) surfer

• either surf a random webpage with probability $\alpha$

• or surf a linked webpage with probability $1 - \alpha$

$\alpha$ : the jumping constant

$$p = \alpha\left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right) + (1 - \alpha)pW$$
Random Walks: PageRank

Two equivalent ways to define PageRank \( p = pr(\alpha, s) \)

(1) \[ p = \alpha s + (1 - \alpha) pW \]

(2) \[ p = \alpha \sum_{t=0}^{\infty} (1 - \alpha)^t (s W^t) \]

\( s = \left( \frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n} \right) \) → the (original) PageRank

\( s = \text{some “seed”, e.g., } (1, 0, \ldots, 0) \) → personalized PageRank
Flashback: Zeno’s paradox

- Lance Armstrong and the tortoise have a race
- Lance is 10x faster
- Tortoise has a 1m head start at time 0
- So, when Lance gets to 1m the tortoise is at 1.1m
- So, when Lance gets to 1.1m the tortoise is at 1.11m ...
- So, when Lance gets to 1.11m the tortoise is at 1.111m ... and Lance will *never* catch up -?

$$1 + 0.1 + 0.01 + 0.001 + 0.0001 + ... = ?$$

unresolved until calculus was invented
Zeno: powned by telescoping sums

Let \( x \) be less than 1. Then

\[
y = 1 + x + x^2 + x^3 + \ldots + x^n
\]

\[
y(1 - x) = (1 + x + x^2 + x^3 + \ldots + x^n)(1 - x)
\]

\[
y(1 - x) = (1 - x) + (x - x^2) + (x^2 - x^3) + \ldots + (x^n - x^{n+1})
\]

\[
y(1 - x) = 1 - x^{n+1}
\]

\[
y = \frac{1 - x^{n+1}}{(1 - x)}
\]

\[
y \approx (1 - x)^{-1}
\]

Example: \( x = 0.1 \), and \( 1 + 0.1 + 0.01 + 0.001 + \ldots \), \( 1.11111 = 10/9 \).
Graph = Matrix
Vector = Node $\rightarrow$ Weight

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>_</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>_</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>_</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>_</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td>1</td>
<td>_</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>_</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td>_</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td>_</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>_</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>_</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|   | A |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|
| A | 3 |   |   |   |   |   |   |   |   |
| B | 2 |   |   |   |   |   |   |   |   |
| C | 3 |   |   |   |   |   |   |   |   |
| D |   |   |   |   |   |   |   |   |   |
| E |   |   |   |   |   |   |   |   |   |
| F |   |   |   |   |   |   |   |   |   |
| G |   |   |   |   |   |   |   |   |   |
| H |   |   |   |   |   |   |   |   |   |
| I |   |   |   |   |   |   |   |   |   |
| J |   |   |   |   |   |   |   |   |   |
Racing through a graph?

Let $W[i,j]$ be $\Pr(\text{walk to } j \text{ from } i)$ and let $\alpha$ be less than 1. Then:

$$Y = I + \alpha W + (\alpha W)^2 + (\alpha W)^3 + ... (\alpha W)^n$$

$$Y(I - \alpha W) = (I + \alpha W + (\alpha W)^2 + (\alpha W)^3 + ...)(I - \alpha W)$$

$$Y(I - \alpha W) = (I - \alpha W) + (\alpha W - (\alpha W)^2 + ...) (I - \alpha W)$$

$$Y(I - \alpha W) = I - (\alpha W)^{n+1}$$

$$Y \approx (I - \alpha W)^{-1} \quad \text{Y}[i,j] = \frac{1}{Z} \Pr(j \mid i)$$

The matrix $(I - \alpha W)$ is the Laplacian of $\alpha W$.

Generally the Laplacian is $(D - A)$ where $D[i,i]$ is the degree of $i$ in the adjacency matrix $A$. 
Random Walks: PageRank

Two equivalent ways to define PageRank $p = pr(\alpha, s)$

1. $\quad p = \alpha s + (1 - \alpha) pW$

2. $\quad p = \alpha \sum_{t=0}^{\infty} (1 - \alpha)^t (sW^t)$

$s = \left( \frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n} \right)$ \quad \rightarrow \quad \text{the (original) PageRank}$

$s = \text{some "seed", e.g., } (1, 0, \ldots, 0)$ \quad \rightarrow \quad \text{personalized PageRank}$
Approximate PageRank: Key Idea

By definition PageRank is fixed point of:

Claim:

Proof:

define a matrix for the pr operator:

\[ R_\alpha s = \text{pr}(\alpha, s) \]
Approximate PageRank: Key Idea

By definition PageRank is fixed point of:

\[ pr(\alpha, s) = \alpha s + (1 - \alpha) pr(\alpha, s) W, \]

\[ pr(\alpha, s) = \alpha s + (1 - \alpha) pr(\alpha, sW). \]

Claim:

Proof:

\[ R_\alpha = \alpha \sum_{t=0}^{\infty} (1 - \alpha)^t W^t \]

\[ = \alpha I + (1 - \alpha) W R_\alpha. \]

\[ pr(\alpha, s) = s R_\alpha \]

\[ = \alpha s + (1 - \alpha) sW R_\alpha \]

\[ = \alpha s + (1 - \alpha) pr(\alpha, sW). \]
Approximate PageRank: Key Idea

By definition PageRank is fixed point of:

\[ \text{pr}(\alpha, s) = \alpha s + (1 - \alpha)\text{pr}(\alpha, s)W, \]

Claim:

\[ \text{pr}(\alpha, s) = \alpha s + (1 - \alpha)\text{pr}(\alpha, sW). \]

Recursively compute PageRank of “neighbors of \( s \)” (=sW), then adjust

Key idea in apr:

- do this “recursive step” repeatedly
- focus on nodes where finding PageRank from neighbors will be useful
Approximate PageRank: Key Idea

\[ \text{pr}(\alpha, s) = \alpha s + (1 - \alpha)\text{pr}(\alpha, sW). \quad W = \frac{I + P}{2} \]

**push**\(_u\)(\(p, r\)):

1. Let \(p' = p\) and \(r' = r\), except for the following changes:
   
   (a) \(p'(u) = p(u) + \alpha r(u)\).
   
   (b) \(r'(u) = (1 - \alpha)r(u)/2\).
   
   (c) For each \(v\) such that \((u, v) \in E\):
       \[ r'(v) = r(v) + (1 - \alpha)r(u)/(2d(u)). \]

2. Return \((p', r')\).

- \(p\) is current approximation (start at 0)
- \(r\) is set of “recursive calls to make”
  - residual error
  - start with all mass on \(s\)
- \(u\) is the node picked for the next call
Lemma 1. Let $p'$ and $r'$ be the result of the operation $\text{push}_u$ on $p$ and $r$. Then

$$p' + \text{pr}(\alpha, r') = p + \text{pr}(\alpha, r).$$

Proof of Lemma 1. After the push operation, we have

$$p' = p + \alpha r(u) \chi_u.$$

$$r' = r - r(u) \chi_u + (1 - \alpha) r(u) \chi_u W.$$

Using equation (5),

$$p + \text{pr}(\alpha, r) = p + \text{pr}(\alpha, r - r(u) \chi_u) + \text{pr}(\alpha, r(u) \chi_u)$$

$$= p + \text{pr}(\alpha, r - r(u) \chi_u) + [\alpha r(u) \chi_u + (1 - \alpha) \text{pr}(\alpha, r(u) \chi_u W)]$$

$$= [p + \alpha r(u) \chi_u] + \text{pr}(\alpha, [r - r(u) \chi_u + (1 - \alpha) r(u) \chi_u W])$$

$$= p' + \text{pr}(\alpha, r').$$

Using the linearity property,

$$\text{pr}(\alpha, r - r(u) \chi_u) + (1-\alpha) \text{pr}(\alpha, r(u) \chi_u W) = \text{pr}(\alpha, r - r(u) \chi_u + (1-\alpha) r(u) \chi_u W)$$

$$\text{pr}(\alpha, s) = \alpha s + (1 - \alpha) \text{pr}(\alpha, sW).$$

(5)
Approximate PageRank: Algorithm

**ApproximatePageRank** \((v, \alpha, \epsilon)\):

1. Let \(p = \vec{0}\), and \(r = \chi_v\).

2. While \(\max_{u \in V} \frac{r(u)}{d(u)} \geq \epsilon\):
   
   (a) Choose any vertex \(u\) where \(\frac{r(u)}{d(u)} \geq \epsilon\).
   
   (b) Apply \(\text{push}_u\) at vertex \(u\), updating \(p\) and \(r\).

3. Return \(p\), which satisfies \(p = \text{apr}(\alpha, \chi_v, r)\) with \(\max_{u \in V} \frac{r(u)}{d(u)} < \epsilon\).

**push\(_u\)\((p, r)\):**

1. Let \(p' = p\) and \(r' = r\), except for the following changes:
   
   (a) \(p'(u) = p(u) + \alpha r(u)\).
   
   (b) \(r'(u) = (1 - \alpha)r(u)/2\).

   (c) For each \(v\) such that \((u, v) \in E\): \(r'(v) = r(v) + (1 - \alpha)r(u)/(2d(u))\).

2. Return \((p', r')\).
Analysis

Lemma 1. Let $p'$ and $r'$ be the result of the operation $\text{push}_u$ on $p$ and $r$. Then

$$p' + \text{pr}(\alpha, r') = p + \text{pr}(\alpha, r).$$

So, at every point in the $apr$ algorithm:

$$p + \text{pr}(\alpha, r) = \text{pr}(\alpha, \chi_v),$$

Also, at each point, $|r|_1$ decreases by $\alpha \cdot \epsilon \cdot \text{degree}(u)$, so:

after $T$ push operations where $\text{degree}(\text{i-th } u) = d_i$, we know

$$\sum_i d_i \cdot \alpha \epsilon \leq 1 \quad \Rightarrow \quad \sum_{i=1}^T d_i \leq \frac{1}{\epsilon \alpha}.$$
Theorem 1. ApproximatePageRank$(v, \alpha, \epsilon)$ runs in time $O(\frac{1}{\epsilon \alpha})$, and computes an approximate PageRank vector $p = \text{apr}(\alpha, \chi_v, r)$ such that the residual vector $r$ satisfies $\max_{u \in V} \frac{r(u)}{d(u)} < \epsilon$, and such that $\text{vol}(\text{Supp}(p)) \leq \frac{1}{\epsilon \alpha}$.

With the invariant: $p + \text{pr}(\alpha, r) = \text{pr}(\alpha, \chi_v)$,

This bounds the error of $p$ relative to the PageRank vector.
ApproximatePageRank \((v, \alpha, \epsilon)\):

1. Let \(p = \tilde{0}\), and \(r = \chi_v\).
2. While \(\max_{u \in V} \frac{r(u)}{d(u)} \geq \epsilon\):
   
   (a) Choose any vertex \(u\) where \(\frac{r(u)}{d(u)} \geq \epsilon\).
   
   (b) Apply \(\text{push}_u\) at vertex \(u\), updating \(p\) and \(r\).
3. Return \(p\), which satisfies \(p = \text{apr}(\alpha, \chi_v, r)\) with \(\max_{u \in V} \frac{r(u)}{d(u)} < \epsilon\).

\text{push}_u(p, r):

1. Let \(p' = p\) and \(r' = r\), except for the following changes:
   
   (a) \(p'(u) = p(u) + \alpha r(u)\).
   
   (b) \(r'(u) = (1 - \alpha) r(u)/2\).
   
   (c) For each \(v\) such that \((u, v) \in E\): \(r'(v) = r(v) + (1 - \alpha) r(u)/(2d(u))\).
2. Return \((p', r')\).
ApproximatePageRank \((v, \alpha, \epsilon)\):

1. Let \(p = \vec{0}\), and \(r = \chi_v\).
2. While \(\max_{u \in V} \frac{r(u)}{d(u)} \geq \epsilon\):
   
   (a) Choose any vertex \(u\) where \(\frac{r(u)}{d(u)} \geq \epsilon\).
   
   (b) Apply \text{push}_u\) at vertex \(u\), updating \(p\) and \(r\).
3. Return \(p\), which satisfies \(p = \text{apr}(\alpha, \chi_v, r)\) with \(\max_{u \in V} \frac{r(u)}{d(u)} < \epsilon\).

\text{push}_u(p, r):

1. Let \(p' = p\) and \(r' = r\), except for the following changes:
   
   (a) \(p'(u) = p(u) + \alpha r(u)\).
   
   (b) \(r'(u) = (1 - \alpha)r(u)/2\).
   
   (c) For each \(v\) such that \((u, v) \in E\): \(r'(v) = r(v) + (1 - \alpha)r(u)/(2d(u))\).
2. Return \((p', r')\).
Comments – Ordering for Scanning

ApproximatePageRank \((v, \alpha, \epsilon)\):

1. Let \(p = \overline{0}\), and \(r = \chi_v\).
2. While \(\max_{u \in V} \frac{r(u)}{d(u)} \geq \epsilon\):

   Scan repeatedly through an adjacency-list encoding of the graph

   For every line you read \(u, v_1, \ldots, v_{d(u)}\) such that \(r(u)/d(u) > \epsilon\):

   (b) Apply \(\text{push}_u\) at vertex \(u\), updating \(p\) and \(r\).

3. Return \(p\), which satisfies \(p = \text{apr}(\alpha, \chi_v, r)\) with \(\max_{u \in V} \frac{r(u)}{d(u)} < \epsilon\).

benefit: storage is \(O(1/\epsilon \alpha)\) for the hash tables, avoids any seeking
Possible optimizations?

• Much faster than doing random access the first few scans, but then slower the last few
  • ...there will be only a few ‘pushes’ per scan

• Optimizations you might imagine:
  – Parallelize?
  – Hybrid seek/scan:
    • Index the nodes in the graph on the first scan
    • Start seeking when you expect too few pushes to justify a scan
      – Say, less than one push/megabyte of scanning
  – Hotspots:
    • Save adjacency-list representation for nodes with a large \( r(u)/d(u) \) in a separate file of “hot spots” as you scan
    • Then rescan that smaller list of “hot spots” until their score drops below threshold.
Putting this together

• Given a graph
  – that’s too big for memory, and/or
  – that’s only accessible via API
• ...we can extract a *sample* in an interesting area
  – Run the apr/rwr from a seed node
  – Sweep to find a low-conductance subset
• Then
  – compute statistics
  – test out some ideas
  – visualize it...