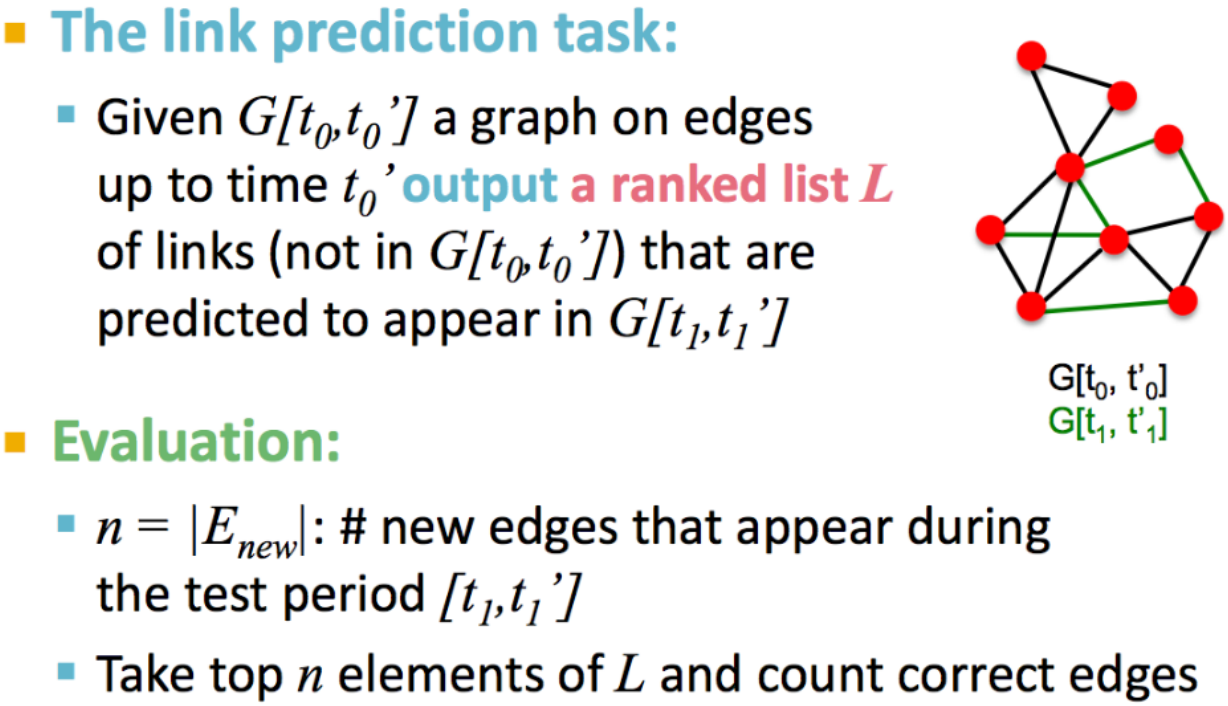
Social Network Analysis: Link Prediction

MIE223
Winter 2025

1 Link Prediction

1.1 Link Prediction in Networks

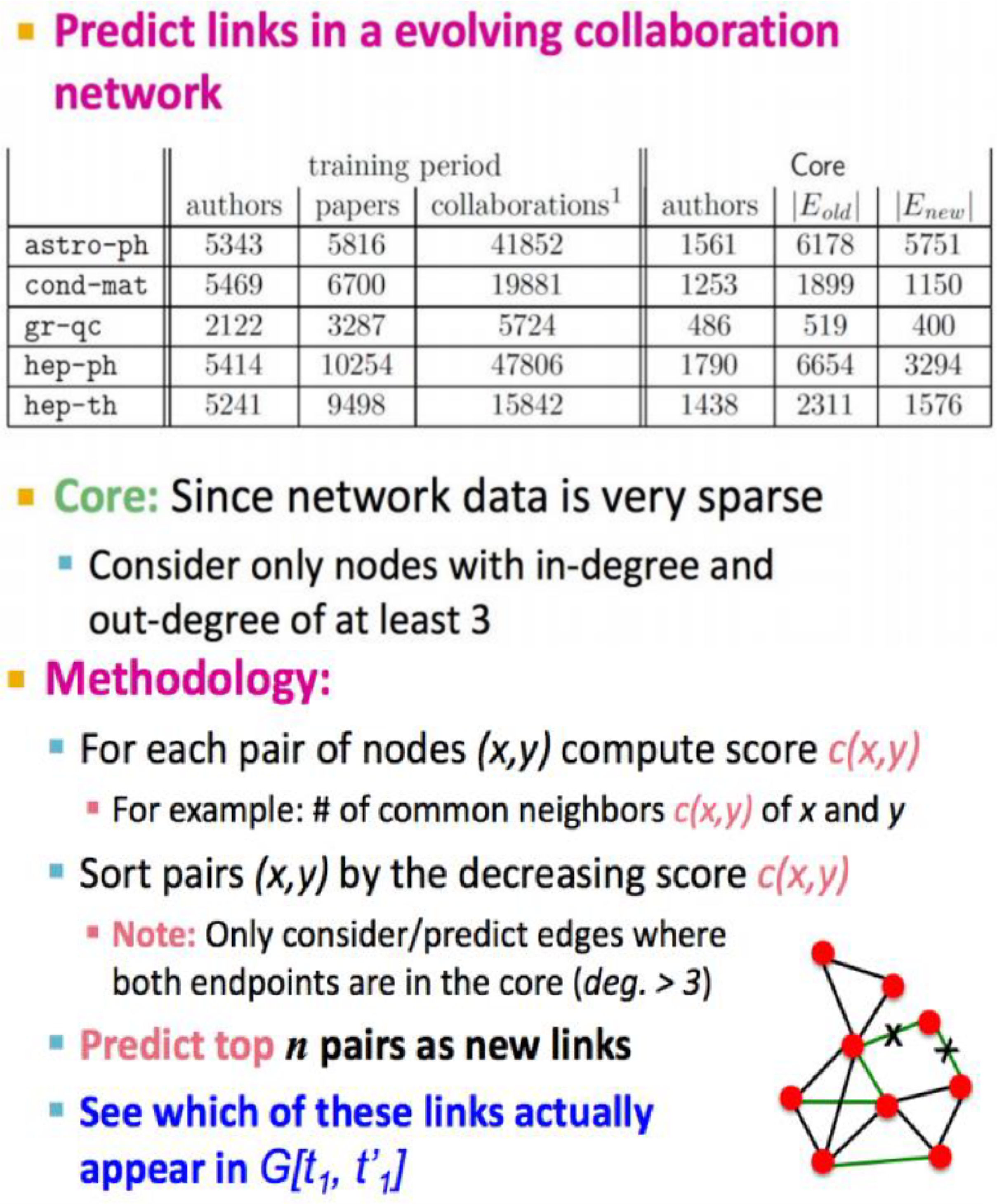


- **The link prediction task:**
  - Given $G[t_0,t_0']$ a graph on edges up to time t_0' **output a ranked list L** of links (not in $G[t_0,t_0']$) that are predicted to appear in $G[t_1,t_1']$

$G[t_0, t'_0]$
$G[t_1, t'_1]$

- **Evaluation:**
  - $n = |E_{new}|$: # new edges that appear during the test period $[t_1,t_1']$
  - Take top n elements of L and count correct edges

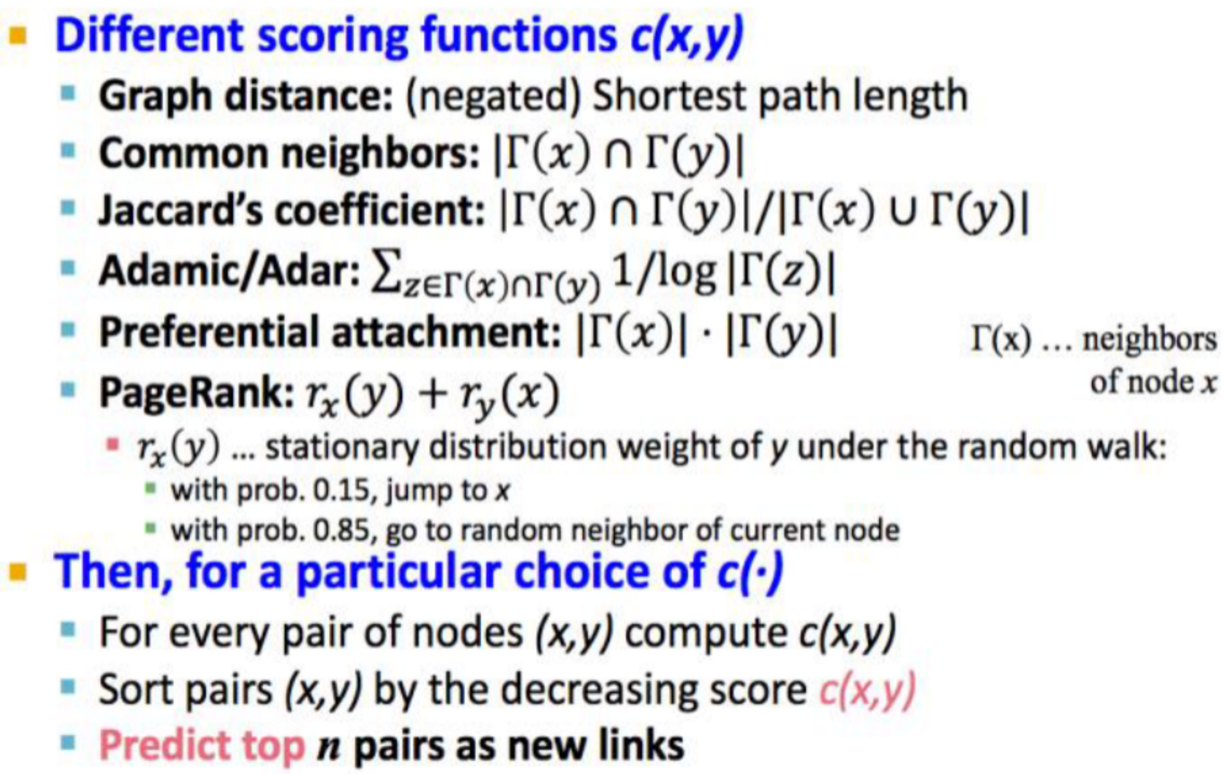
1.2 Link Prediction via Proximity



- **Predict links in a evolving collaboration network**

	training period			Core		
	authors	papers	collaborations[1]	authors	$\lvert E_{old}\rvert$	$\lvert E_{new}\rvert$
astro-ph	5343	5816	41852	1561	6178	5751
cond-mat	5469	6700	19881	1253	1899	1150
gr-qc	2122	3287	5724	486	519	400
hep-ph	5414	10254	47806	1790	6654	3294
hep-th	5241	9498	15842	1438	2311	1576

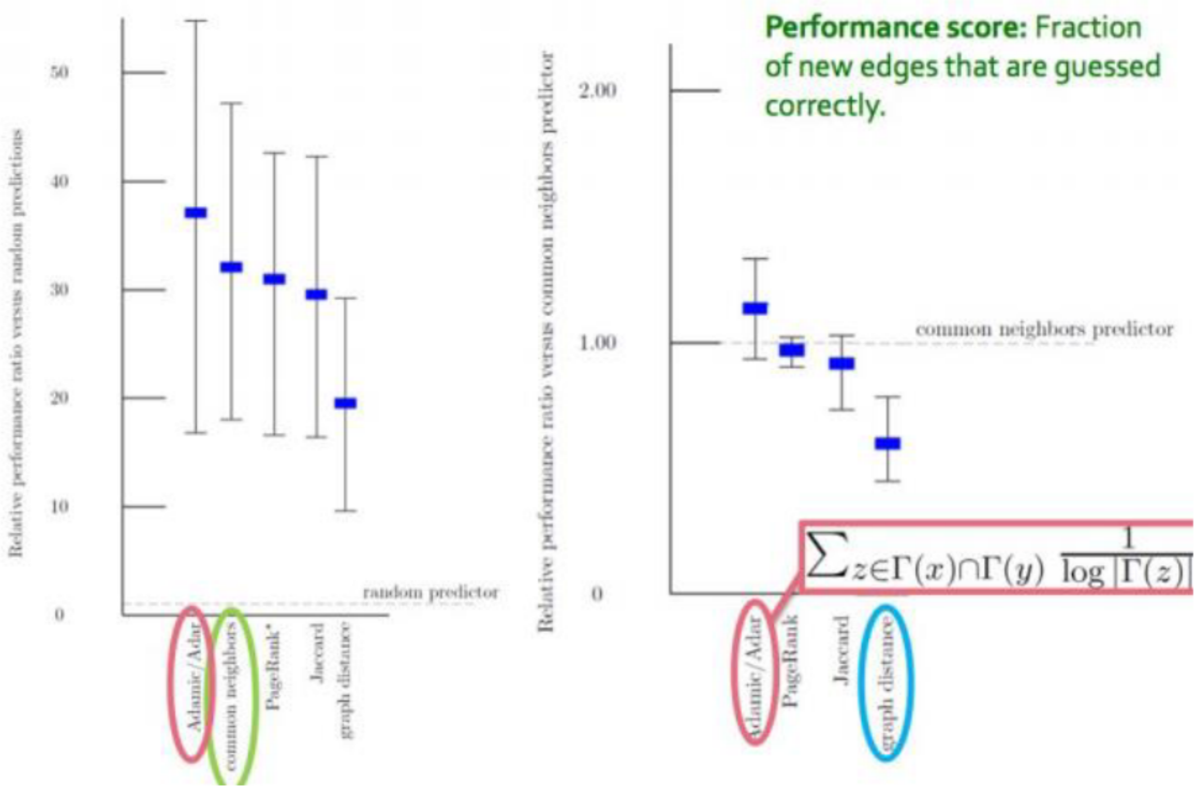
- **Core:** Since network data is very sparse
  - Consider only nodes with in-degree and out-degree of at least 3
- **Methodology:**
  - For each pair of nodes *(x,y)* compute score *c(x,y)*
    - For example: # of common neighbors *c(x,y)* of *x* and *y*
  - Sort pairs *(x,y)* by the decreasing score *c(x,y)*
    - **Note:** Only consider/predict edges where both endpoints are in the core (*deg. > 3*)
  - **Predict top *n* pairs as new links**
  - **See which of these links actually appear in *G[t_1, t'_1]***



- **Different scoring functions *c(x,y)***
  - **Graph distance:** (negated) Shortest path length
  - **Common neighbors:** $|\Gamma(x) \cap \Gamma(y)|$
  - **Jaccard's coefficient:** $|\Gamma(x) \cap \Gamma(y)| / |\Gamma(x) \cup \Gamma(y)|$
  - **Adamic/Adar:** $\sum_{z \in \Gamma(x) \cap \Gamma(y)} 1/\log|\Gamma(z)|$
  - **Preferential attachment:** $|\Gamma(x)| \cdot |\Gamma(y)|$
  - **PageRank:** $r_x(y) + r_y(x)$
    - $r_x(y)$ … stationary distribution weight of *y* under the random walk:
      - with prob. 0.15, jump to *x*
      - with prob. 0.85, go to random neighbor of current node
- **Then, for a particular choice of *c(·)***
  - For every pair of nodes *(x,y)* compute *c(x,y)*
  - Sort pairs *(x,y)* by the decreasing score *c(x,y)*
  - **Predict top *n* pairs as new links**

Γ(x) … neighbors of node *x*

1.3 Results



Performance score: Fraction of new edges that are guessed correctly.