The BTER Graph Model: Blocked Two-Level Erdös-Rényi

C. Seshadri, Tamara G. Kolda, Ali Pinar
Sandia National Labs

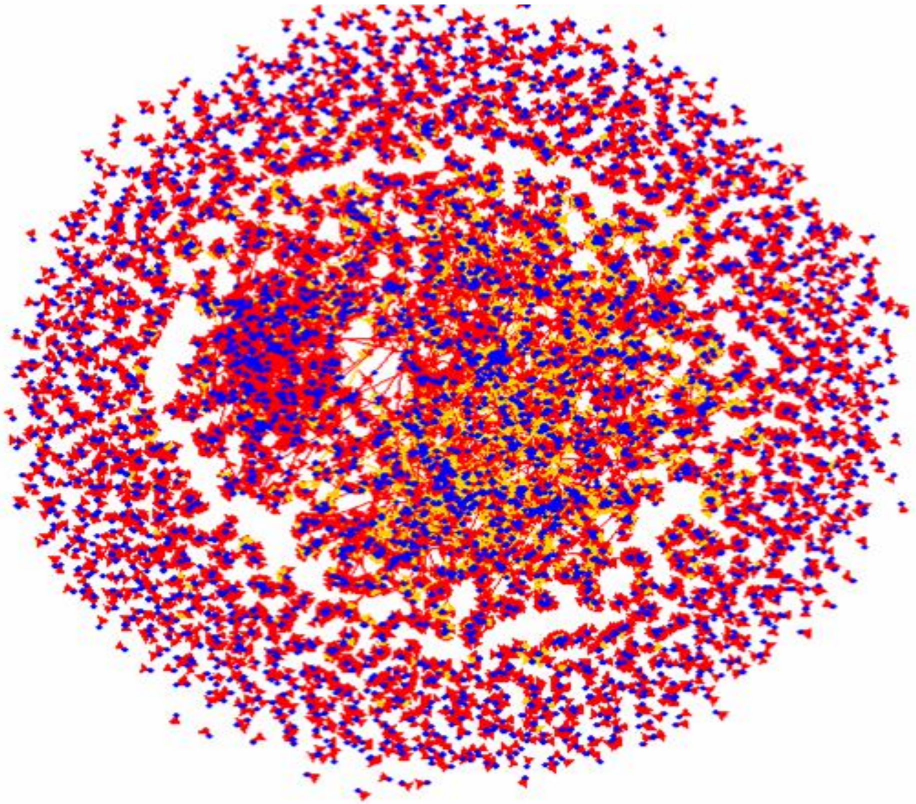
U.S. Department of Energy
Office of Advanced Scientific Computing Research

Thanks to David Gleich for helpful discussions and data, and to Janine Bennett for data preparation.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Why Model Networks?

- Insight into…
  - Generative process
  - Graph properties such as eigenvalue distribution
  - Evolution
- Testing graph algorithms
  - Various scales
  - Various degree distributions
- Enable sharing of realistic but non-sensitive data
  - Computer network traffic
  - Social networks
- Anomaly detection
  - Unusual edges
- Guide statistical sampling



Graph Model Desiderata

- **Goal:** Test graph algorithms
- **Desiderata**
  1. Model a **variety** of "heavy tailed" degree distributions
     - Degree distributions vary heavily between various kinds of graphs (Sala et al., arXiv1108.0027)
  2. High clustering coefficient
     - Ideally, for both low and high degrees nodes
  3. Well-connected
     - Large connected component
     - Small diameter
  4. Scales to large problems
     - 2^{42} nodes and 2^{46} edges for Graph 500

Clustering Coefficient

$$cc_i = \frac{t_i}{\binom{d_i}{2}}$$

t_i = # triangles at vertex i

d_i = degree of vertex i

Global Clustering Coeff.

$$gcc = \frac{\sum_i t_i}{\sum_i \binom{d_i}{2}}$$

Limitations of Current Models

Sala, Cao, Wilson, Zablit, Zheng, Zhao, WWW2010

Inherently Sequential

Feature-driven

- **Barabasi-Albert** –power law deg. dist.
- **Forest Fire** – new node connects to some neighbors of its 1st neighbor and then recurses

Intent-driven

- **Random Walk** – new node's connections depend on random walk from random node in graph
- **Nearest Neighbor** – new node connects to some neighbors of its 1st neighbor

Structure-driven

- **Stochastic Kronecker Graphs** – edges generated via Kronecker product of 2x2 generator matrices
- **dK-graphs** – directly includes subgraph patterns from original graph

Does not Scale

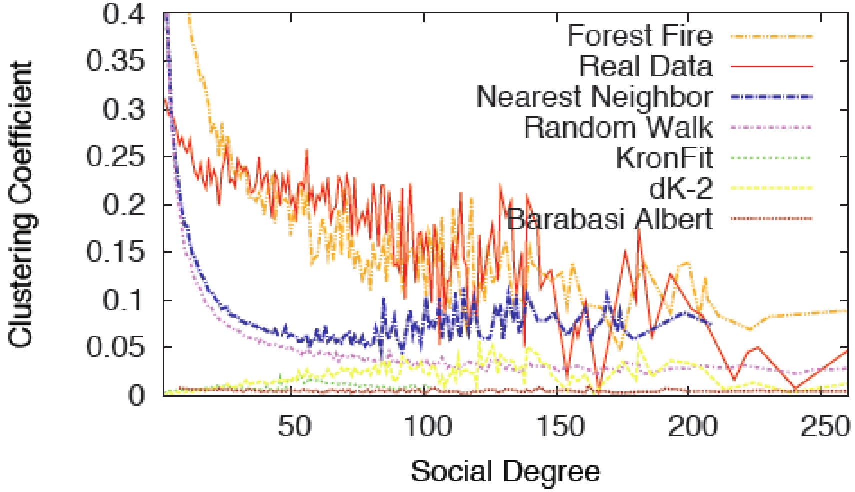


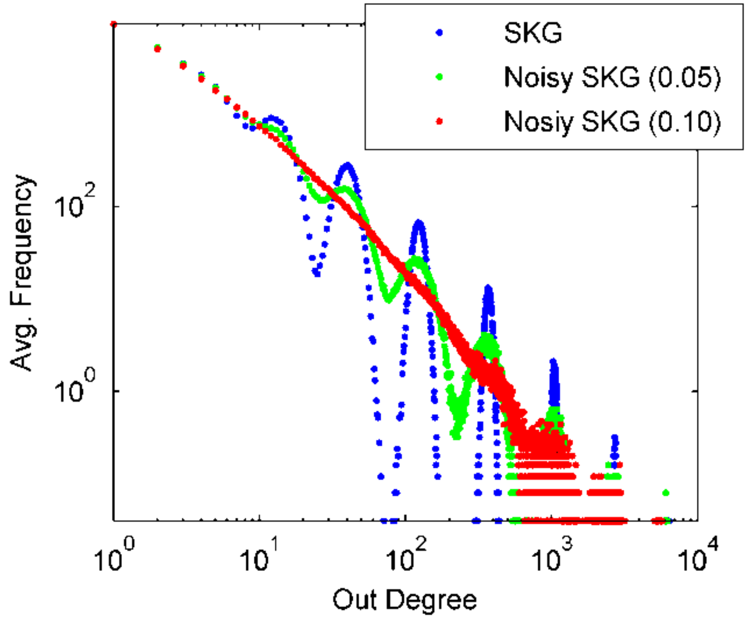
Figure from Sala et al. (2010) showing Santa Barbara facebook social network.

Clearly Best for Scalability,
But Poor Clustering Coefficient

Stochastic Kronecker Graph (SKG): The Model to Beat

Chakrabarti and Faloutsos, SDM04
Leskovec et al., JMLR, 2010
Seshadri, Pinar, Kolda, arXiv: 1102.5046, 2011

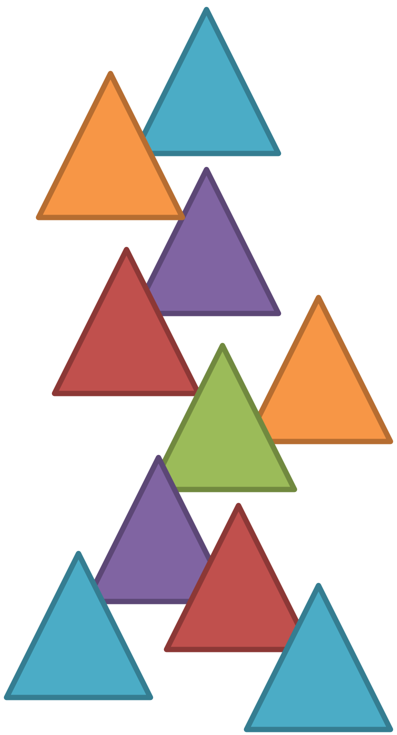
- Generator for Graph500 Supercomputing Benchmark
- PROS
  - Only 4 parameters
  - Very scalable!
- CONS
  - Oscillations in its degree distribution
    - Noisy version fixes problem
  - For Graph 500 parameters, 50-74% of its vertices are isolated
  - Limited degree distributions
  - No community structure



SKG for Graph 500

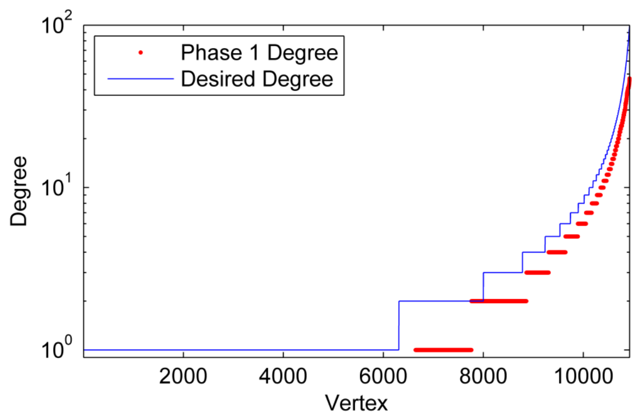
Underlying Principal

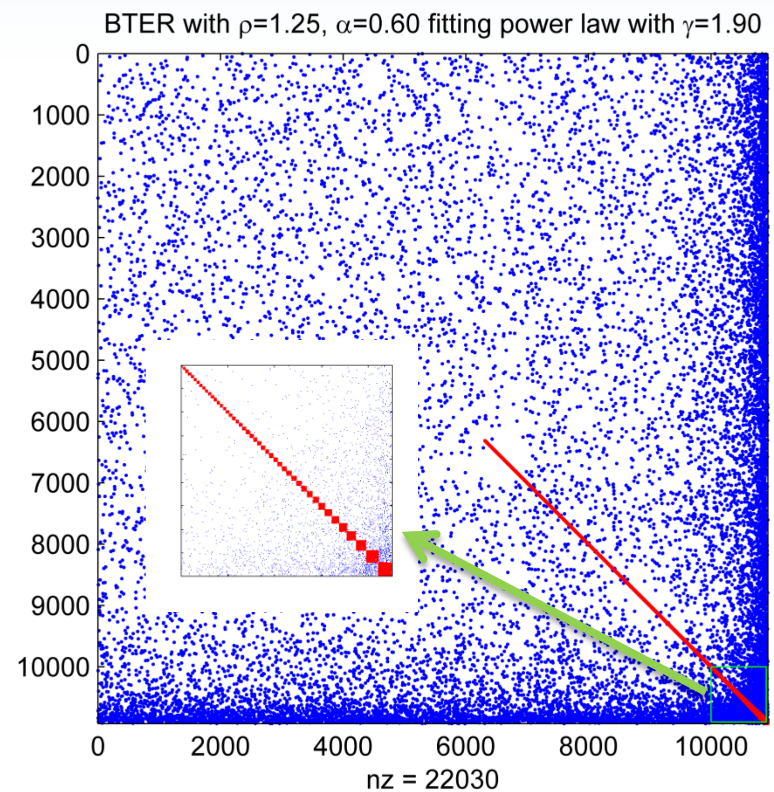
- High clustering coefficients require lots of triangles
  - If (u,v) and (v,w) are edges, probability of (u,w) should be high
- Doesn't occur in any existing non-sequential model since
  - Edges are generated independently
  - Community imposition (e.g. though factor models) is too coarse
- Our idea:
  - Group the nodes together into a large number of small near-cliques
  - Link those groups together randomly



BTER: Block Two-Level ER

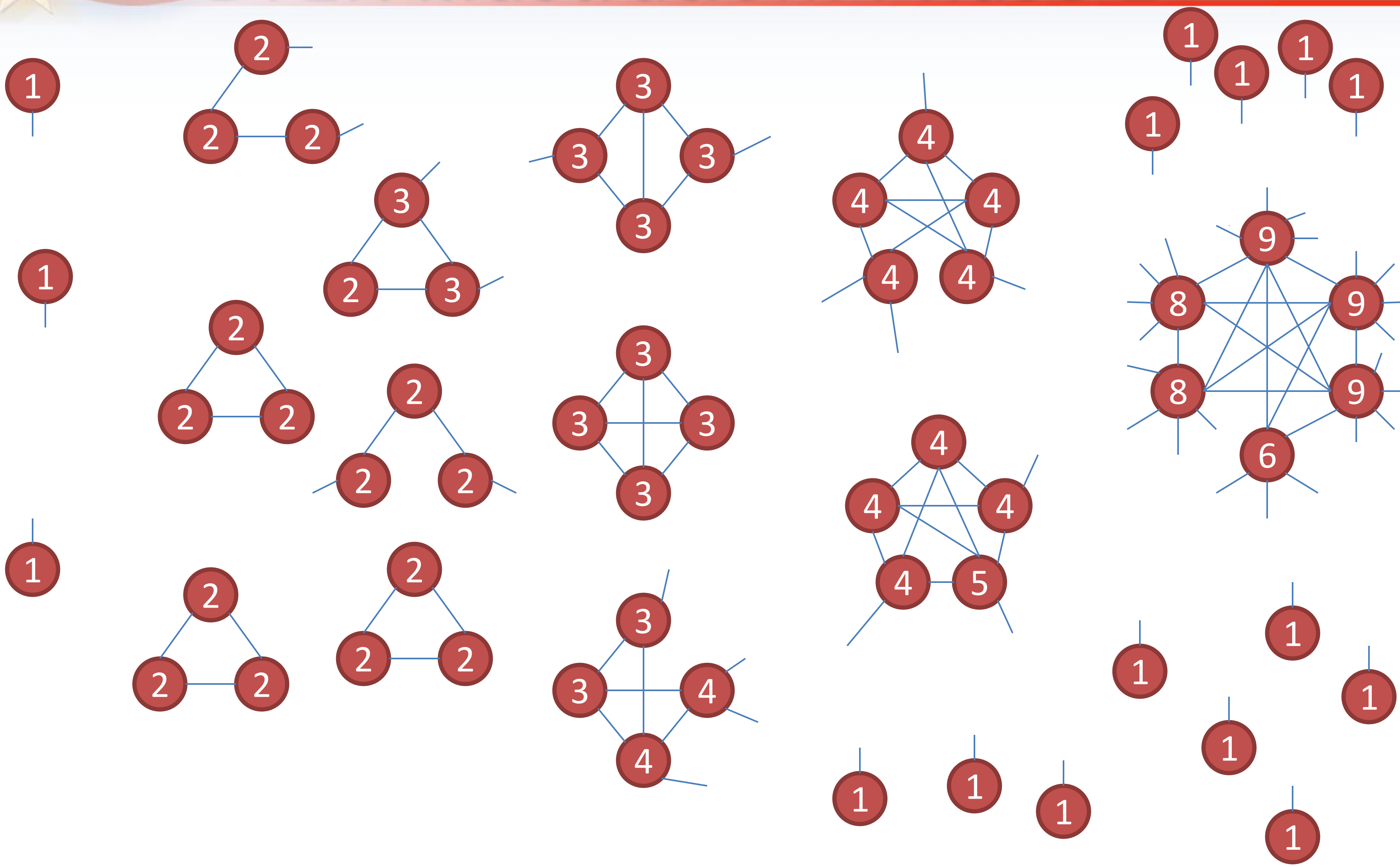
- Phase 1
  - Create near cliques via ER with a high probability such that phase 1 degrees do not exceed desired degrees



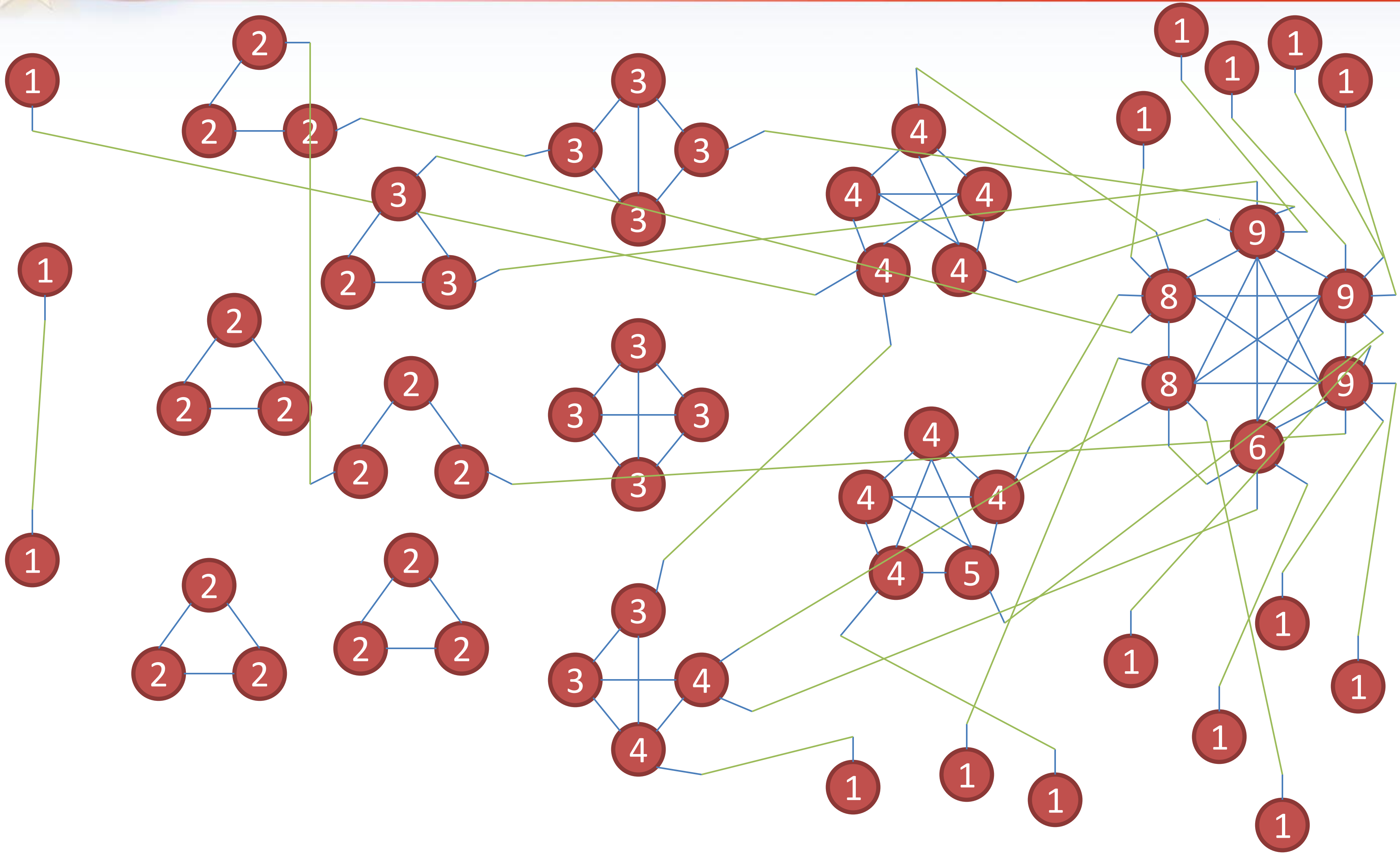


- Phase 2
  - Fill in the remainder of the degree distribution using a weighted ER approach

BTER Illustration: Phase 1



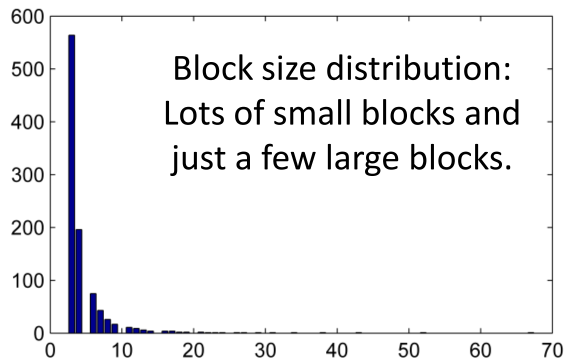
BTER Illustration: Phase 2



BTER Details

Phase 1

- Sort the nodes by degree
- Create blocks
  - v1 = first node in clique
  - v2 = v1 + round(αd(v1))
  - n = v2-v1+1 (*blocksize*)
  - Create an ER-graph of size n with the specified link probability ρ
- Goal of Phase 1 is a high clustering coefficient

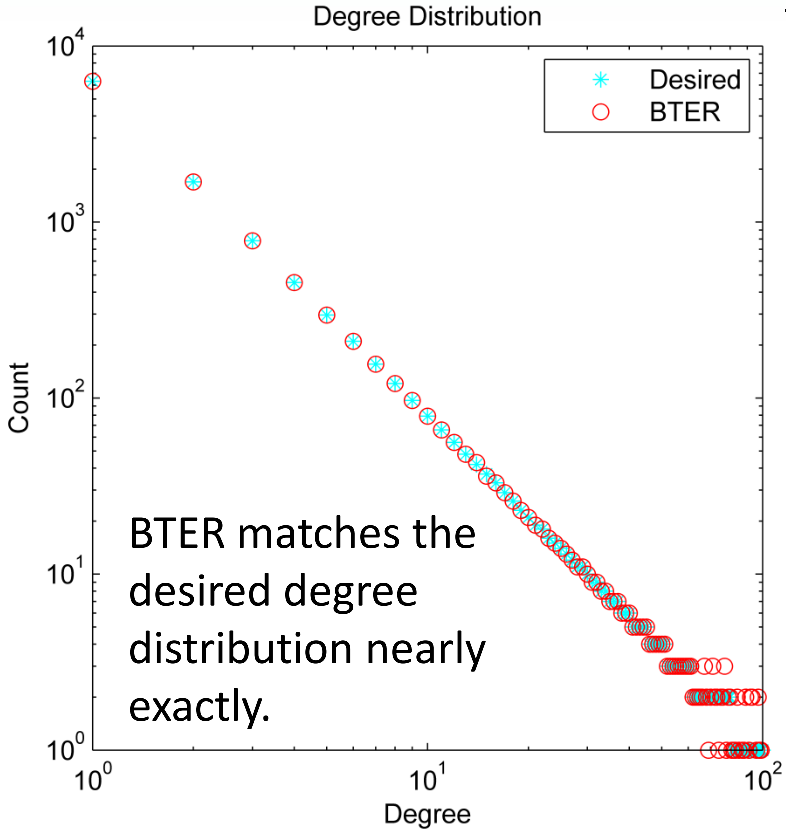


Phase 2

- Creates weighted ER graph to fill in the remaining degrees.
  - Create half-edges for all nodes
  - Randomly match
  - Remove duplicates & self-edges (for both phases)
  - Repeat
- Goal of Phase 2 is matching degree distribution and a low diameter

POWER LAW DEGREE DISTBUTION: PHASE 1 VS PHASE 2

Power Law Degree Distribution



Power Law

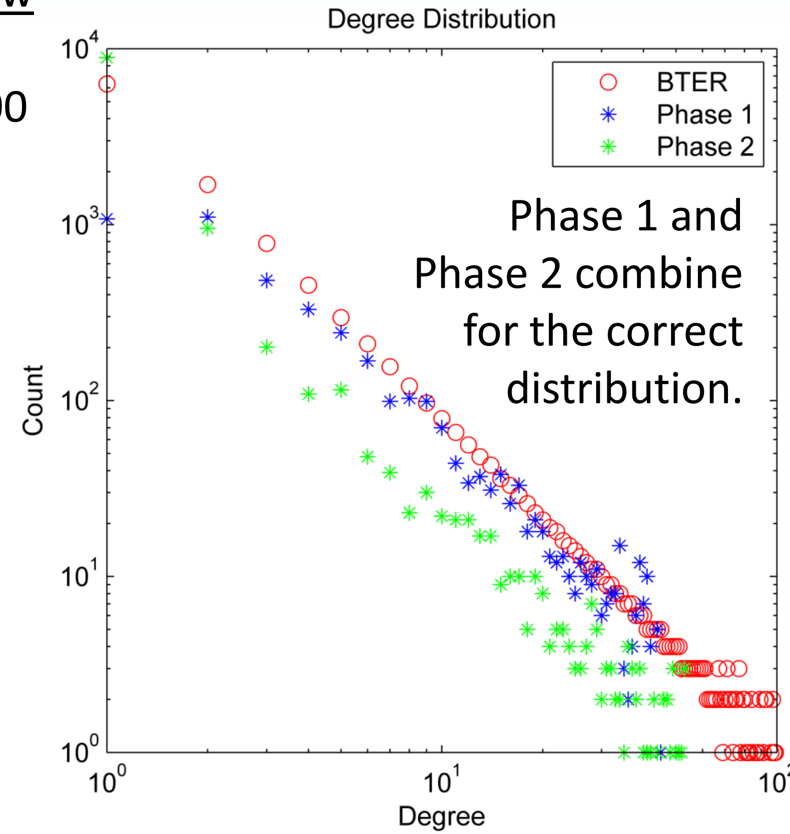
$\gamma = 1.9$

$d_{max} = 100$

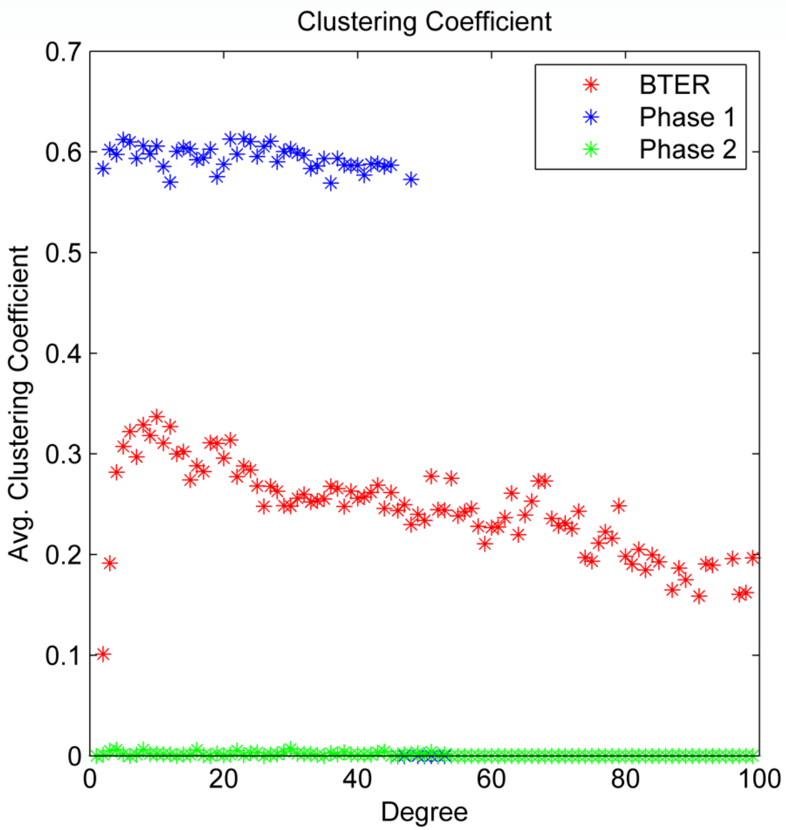
BTER

$\rho = 0.6$

$\alpha = 1.25$



BTER has High Clustering Coefficient

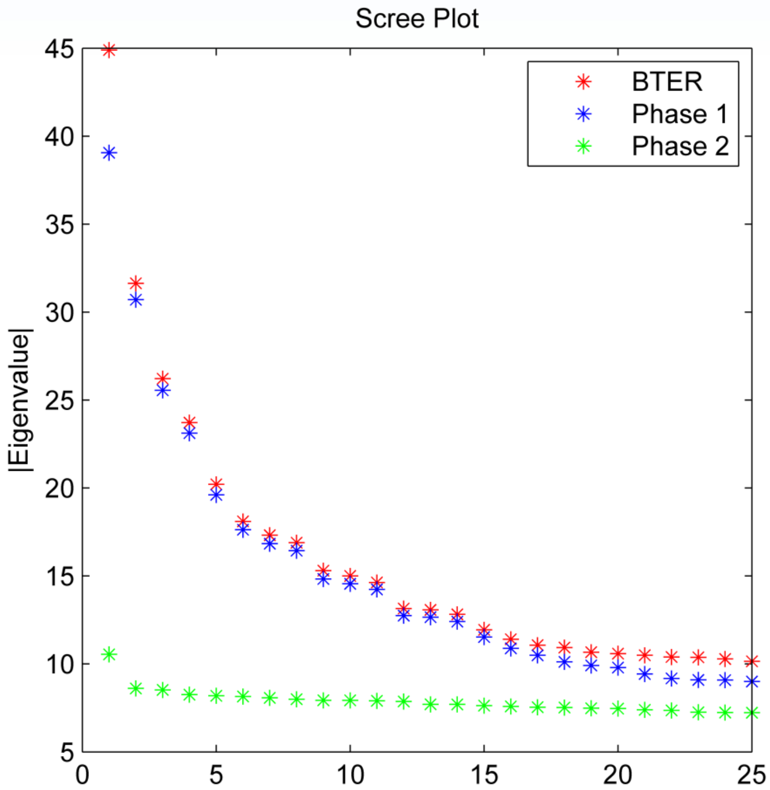


Graph	Nodes	Edges	LCC	DIAM	GCC
BTER	10925	40272	75%	18	0.24
Phase 1	10925	21950	1%	2	0.59
Phase 2	10925	18322	48%	12	0

Note: Diameter is for the LCC and just an upper bound based on 500 random walks.

Eigenvalues Determined by Phase 1

Observe: Eigenvalues of the final BTER model are very close to those of Phase 1.



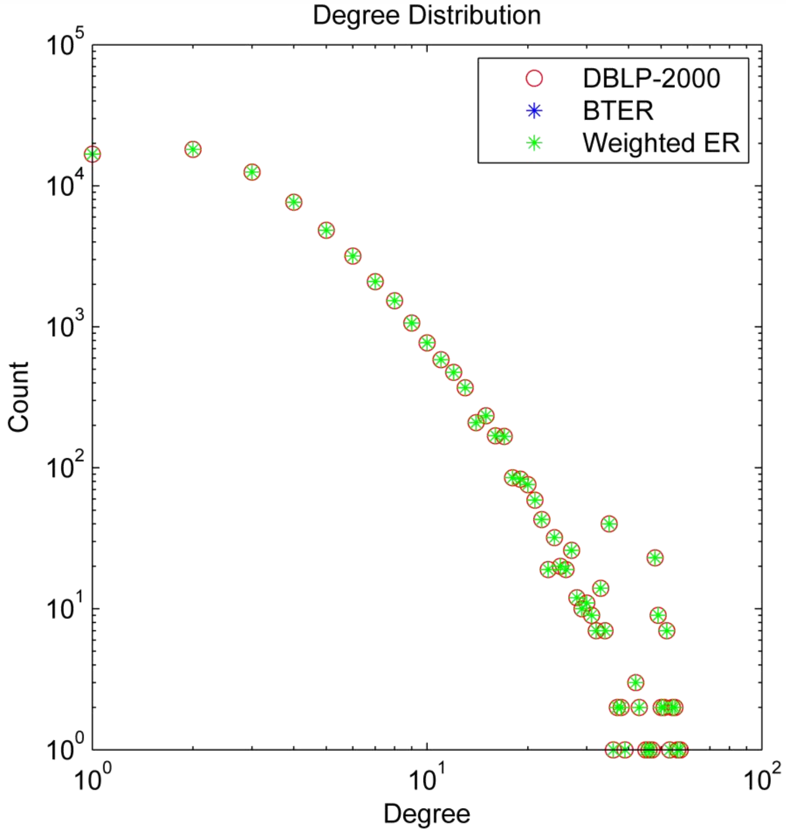
REAL DATA: DBLP CO-AUTHORSHIP

Matching to Real Data: DBLP 2000

<u>DBLP Co-Authors in 2000</u>
71, 390 Authors
253, 908 Links

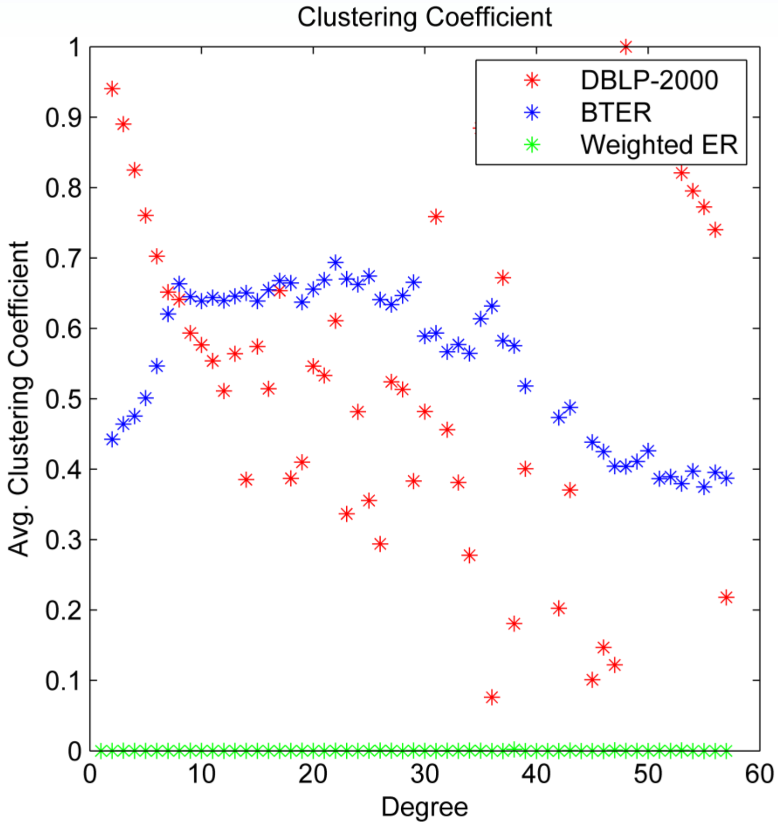
Compare to **Weighted ER**, which does an edge matching to get the desired degree distribution.

Both BTER and Weighted ER match the degree distribution perfectly.



BTER's CC matches DBLP 2000

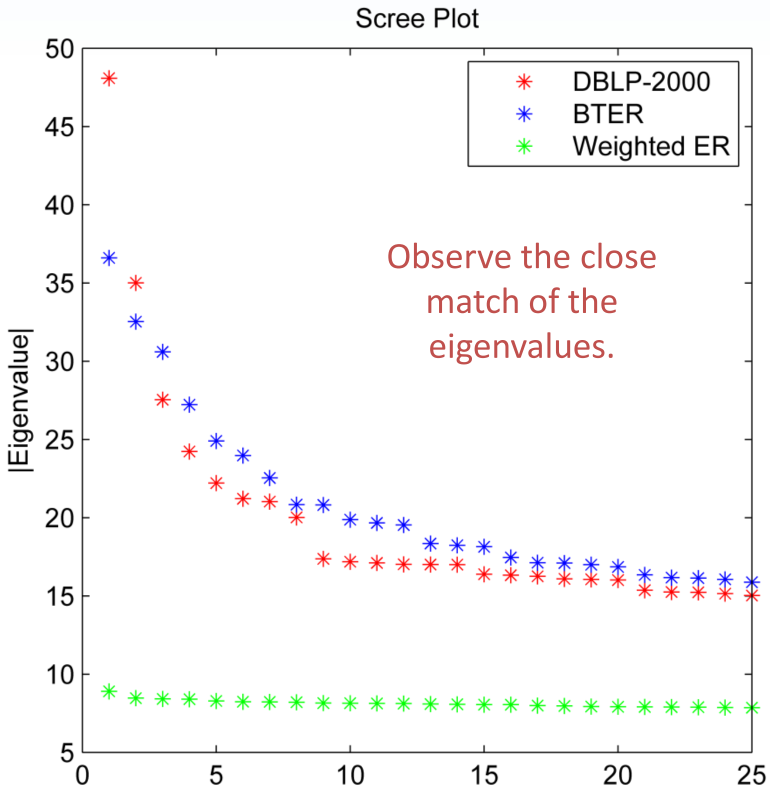
Graph	Nodes	Edges	LCC	DIAM	GCC
DBLP-2000	71389	253908	38%	34	0.65
BTER	71389	253908	73%	60	0.58
Weighted ER	71389	253908	98%	20	0

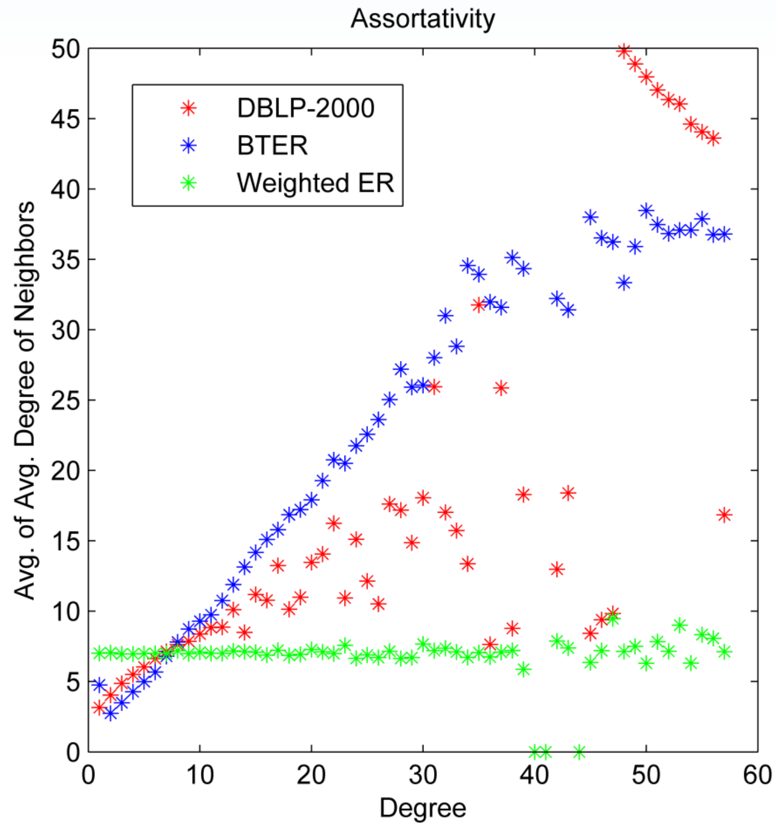


Very close match between real data and BTER in terms of global clustering coefficient (GCC).

BTER
$\rho = 0.8$
$\alpha = 1.15$

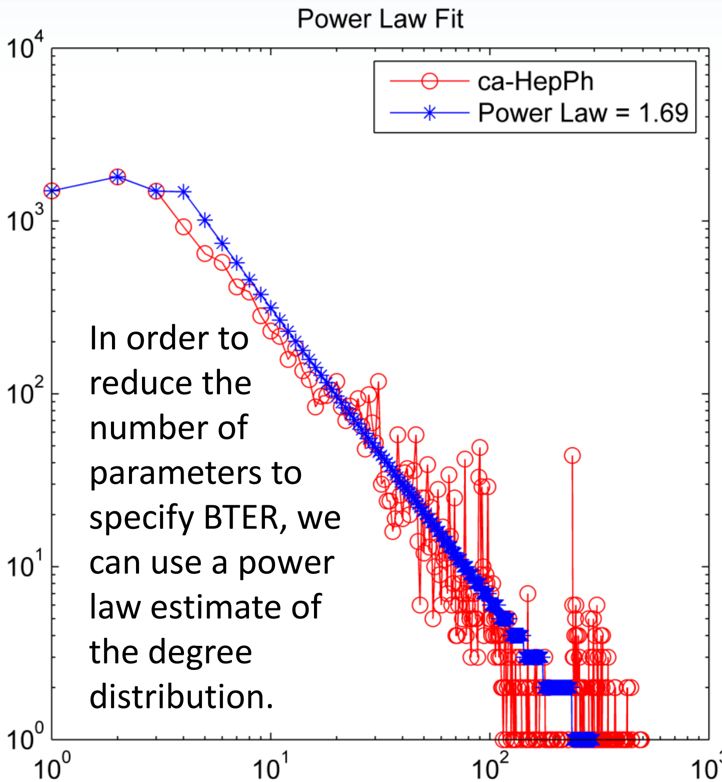
BTER E-vals and Assortativity for DBLP 2000



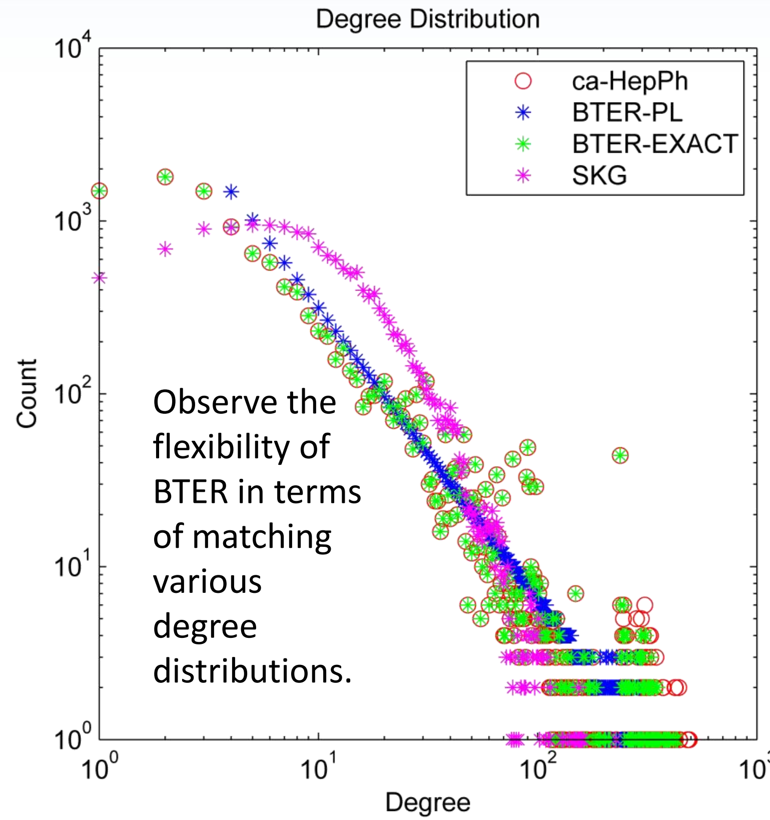


BTER AND SKG ON CA-HEPPH (CO-AUTHORSHIP DATA)

BTER and SKG Comparison: CA-HepPh



In order to reduce the number of parameters to specify BTER, we can use a power law estimate of the degree distribution.



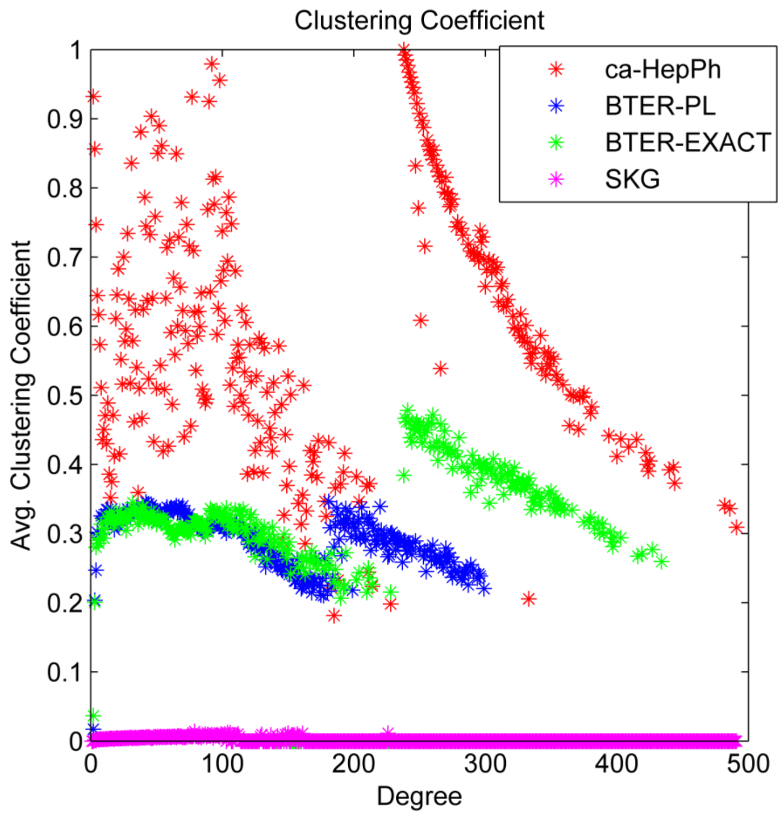
Observe the flexibility of BTER in terms of matching various degree distributions.

Power Law Fit Code from:
A. Clauset, C.R. Shalizi, and M.E.J. Newman, "Power-law distributions in empirical data" *SIAM Review* **51**(4), 661-703 (2009). (doi:10.1137/070710111)

<u>RMAT</u>
T = [0.42, 0.19; 0.19, 0.21]
K=14

<u>BTER</u>
$\rho = 0.6$
$\alpha = 1.25$

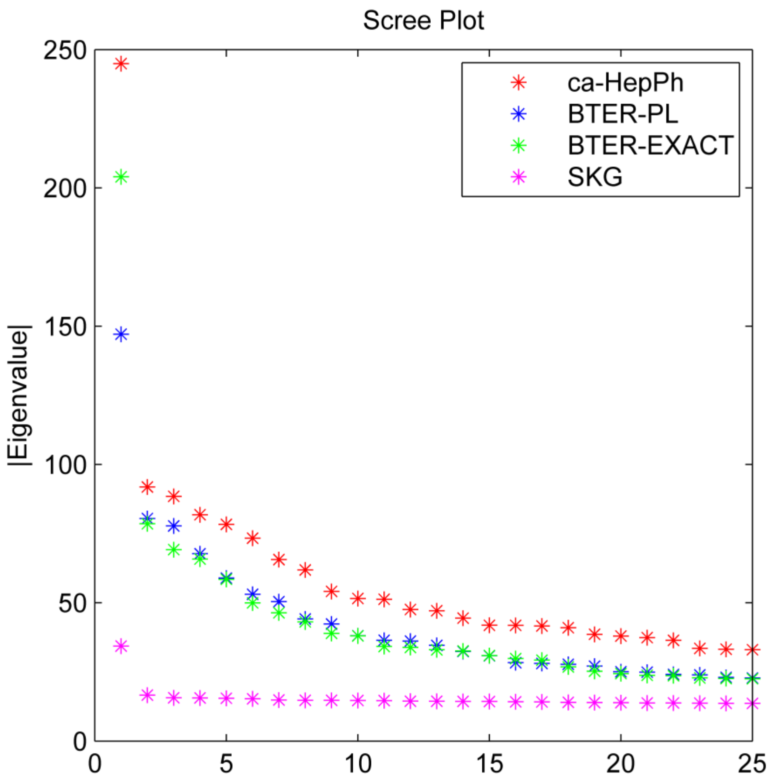
BTER has better clustering coefficients than SKG

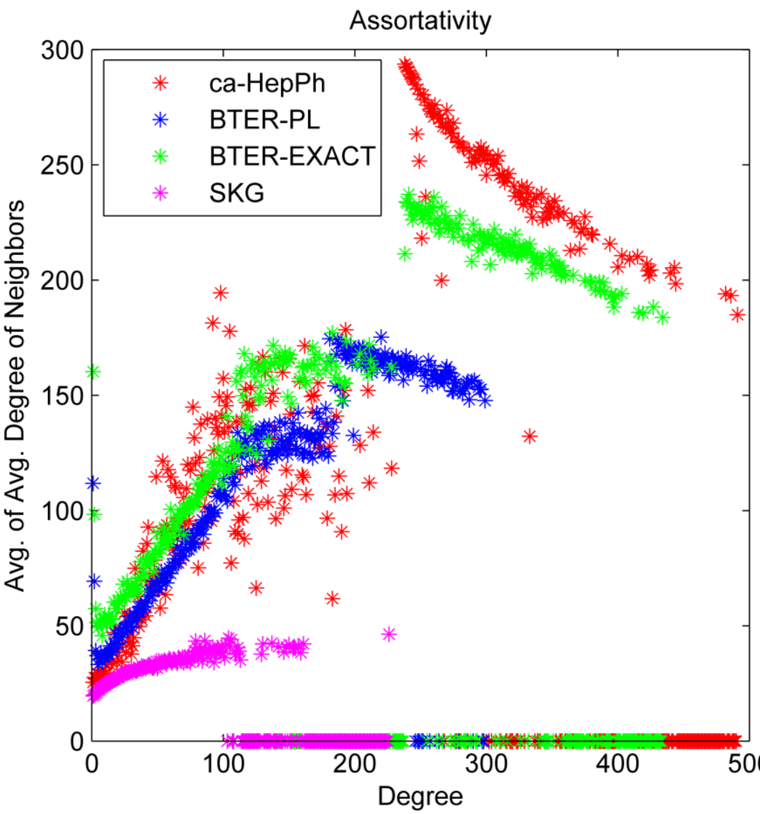


Graph	Nodes	Edges	LCC	DIAM	GCC
ca-HepPh	12008	237010	93%	14	0.66
BTER-PL	13687	225250	100%	10	0.29
BTER-EXACT	12008	235772	100%	10	0.36
SKG	16384	236109	99%	8	0.01

- BTER better than SKG for high CC
  - SKG GCC = 0.01!
- BTER captured behavior in data
  - This was not part of the fitting procedure
  - Note diameter is also a good fit
- Exact degree distribution better than PL estimate

BTER also better in terms of e-val and assortativity for CA-HepPh

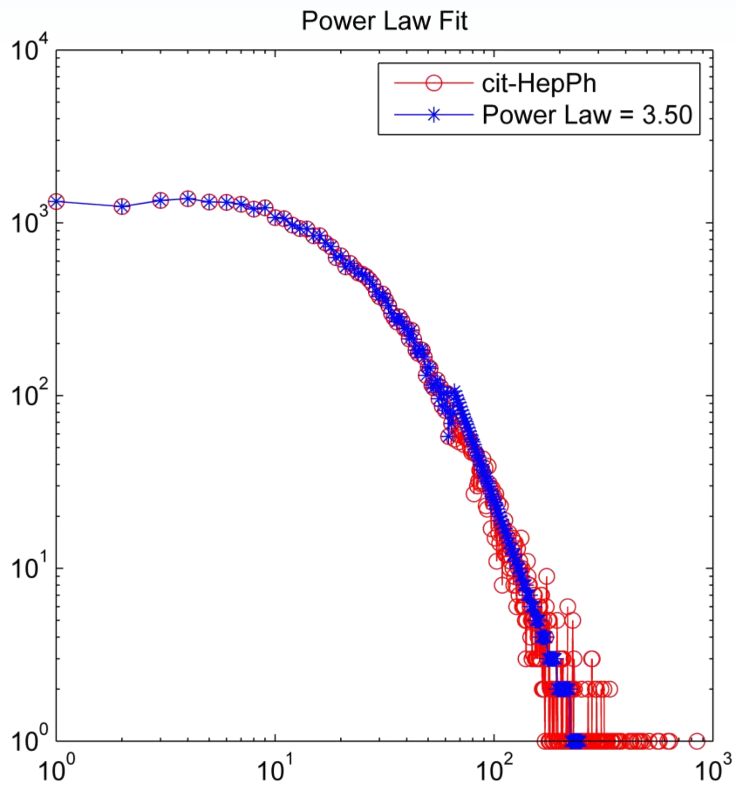


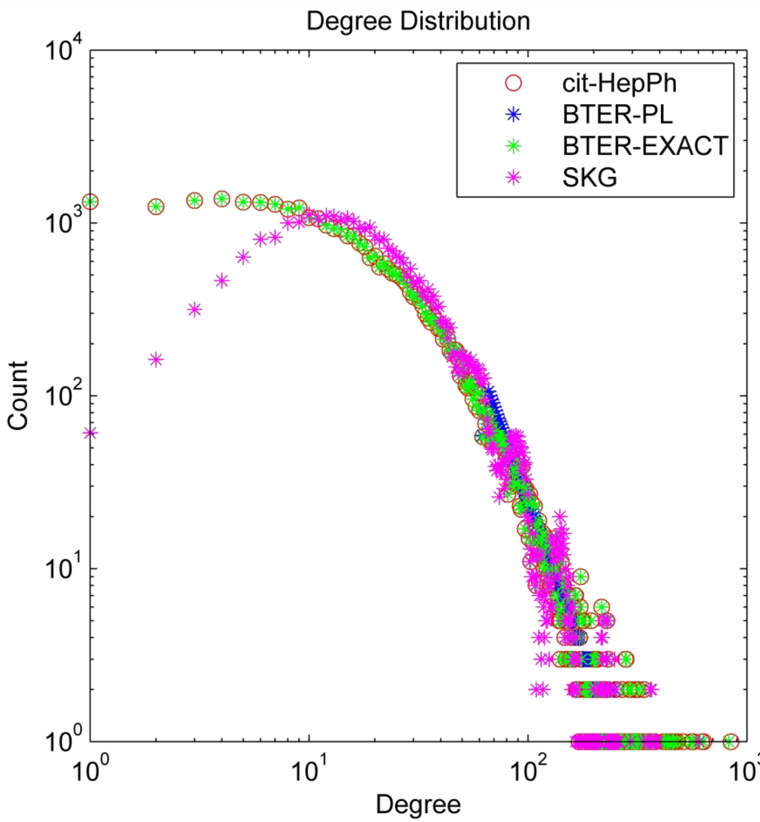


BTER AND SKG ON CIT-HEPPH (CITATION DATA)

BTER compared to SKG on a citation network: CIT-HepPh

We worked with a symmetrized version of this data and the SKG results.





<u>RMAT</u>

T = [0.43, 0.19; 0.15, 0.23]

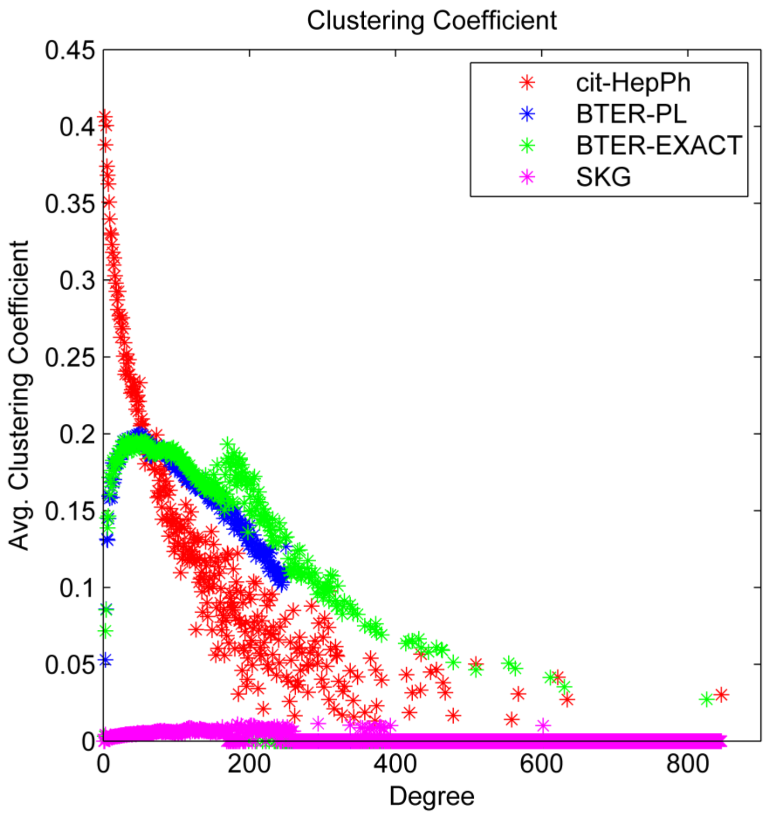
K=14

<u>BTER</u>

ρ = 0.5

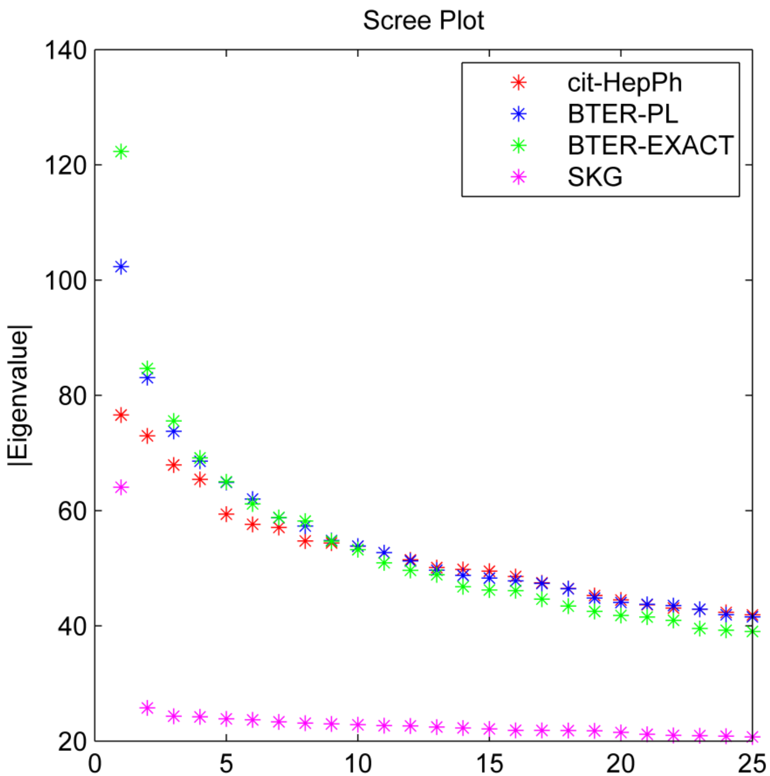
α = 1.25

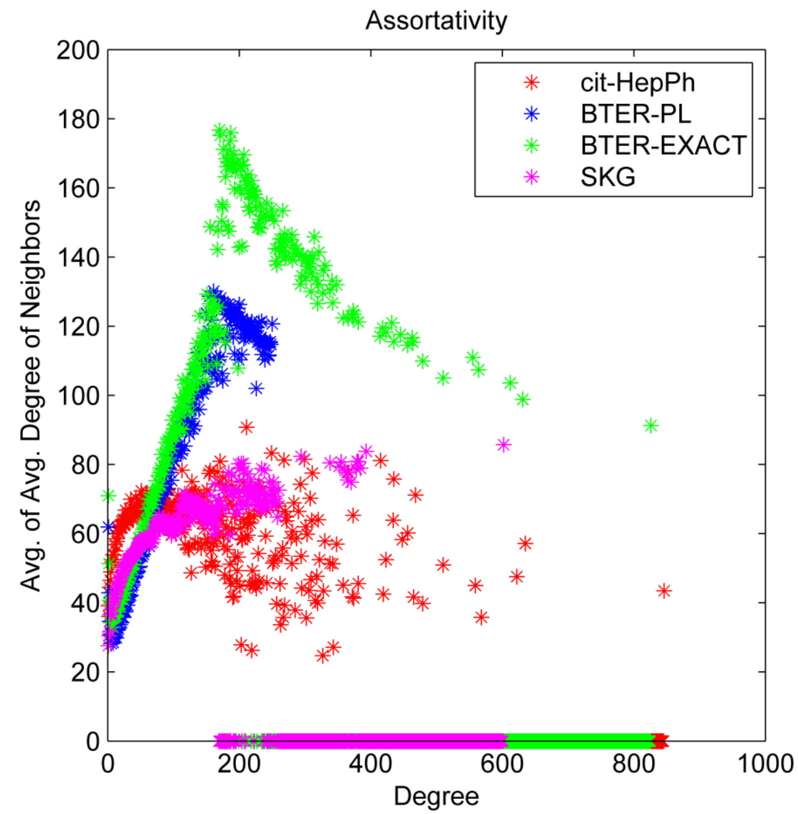
CIT-HepPh Clustering Coeff. Comparison



Graph	Nodes	Edges	LCC	DIAM	GCC
cit-HepPh	34546	841798	100%	12	0.15
BTER-PL	34934	855880	100%	10	0.18
BTER-EXACT	34546	841734	100%	10	0.16
SKG	32768	924017	100%	6	0.01

CIT-HepPh E-vals and Assortativity

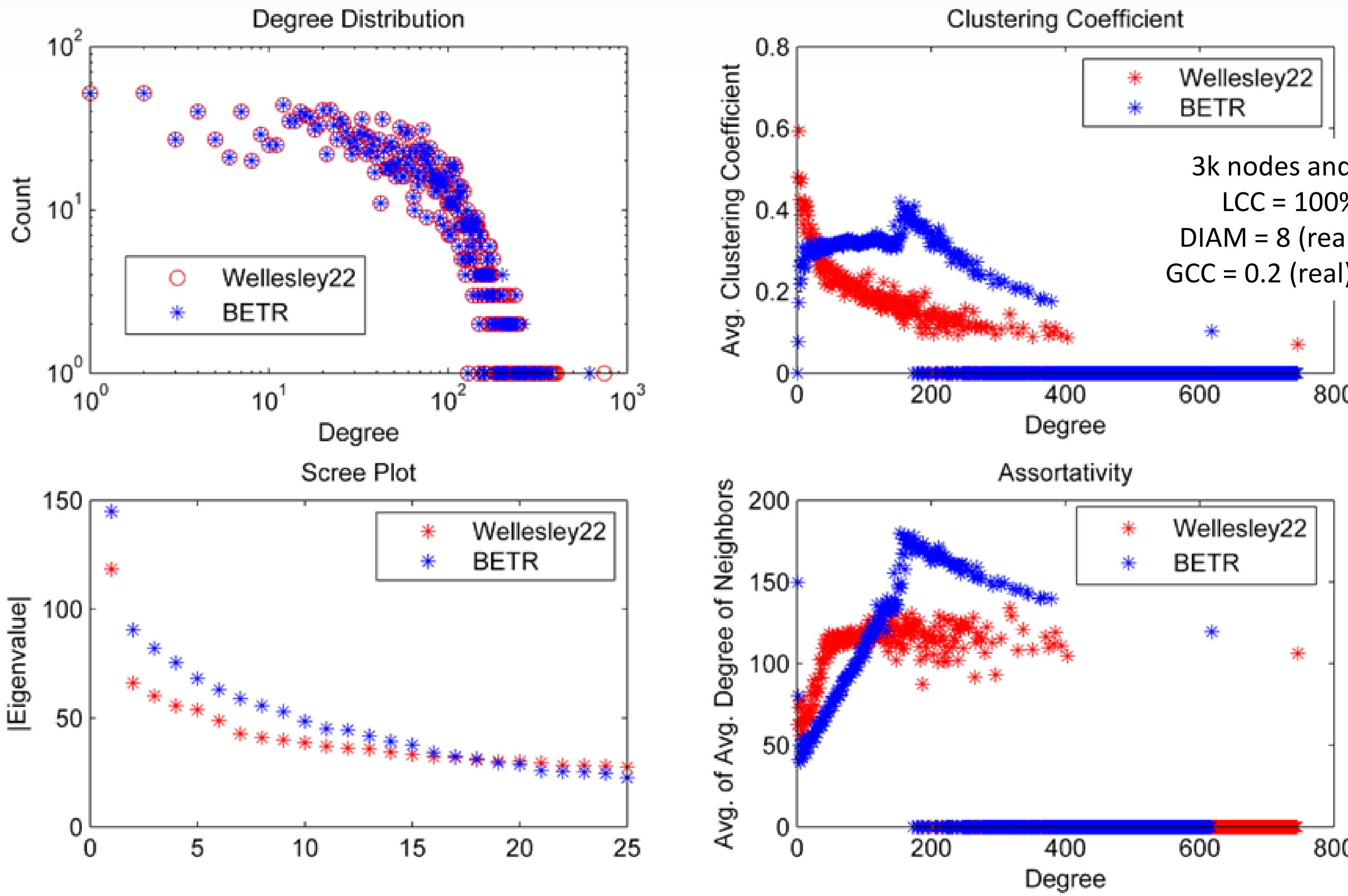




MORE EXAMPLES OF MATCHING REAL-WORLD DATA

Comparison on Social Network

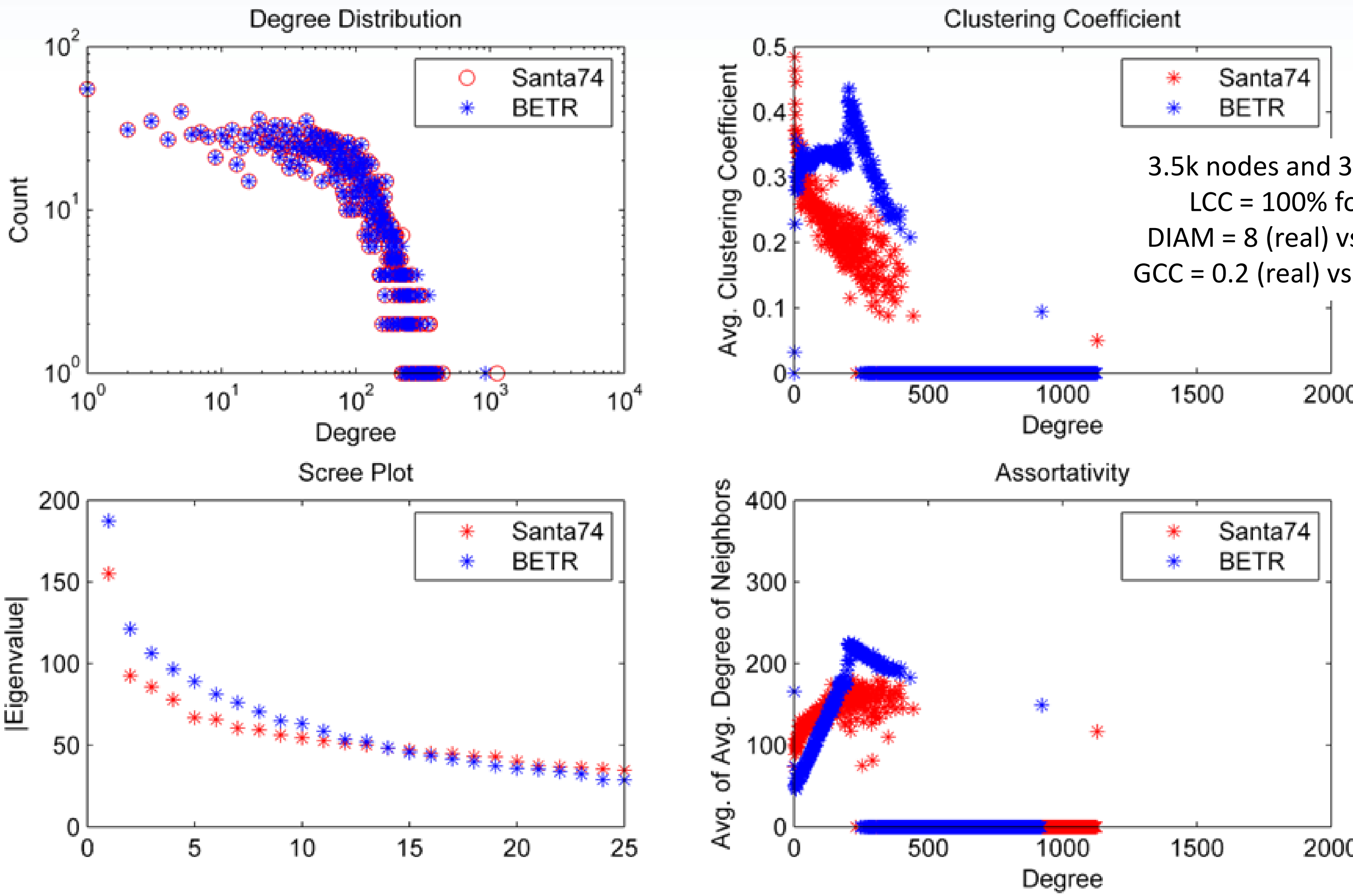
BTER
ρ = 0.6
α = 1.25



3k nodes and 190k edges
LCC = 100% for both
DIAM = 8 (real) vs. 6 (BTER)
GCC = 0.2 (real) vs. 0.3 (BTER)

Comparison on Social Network

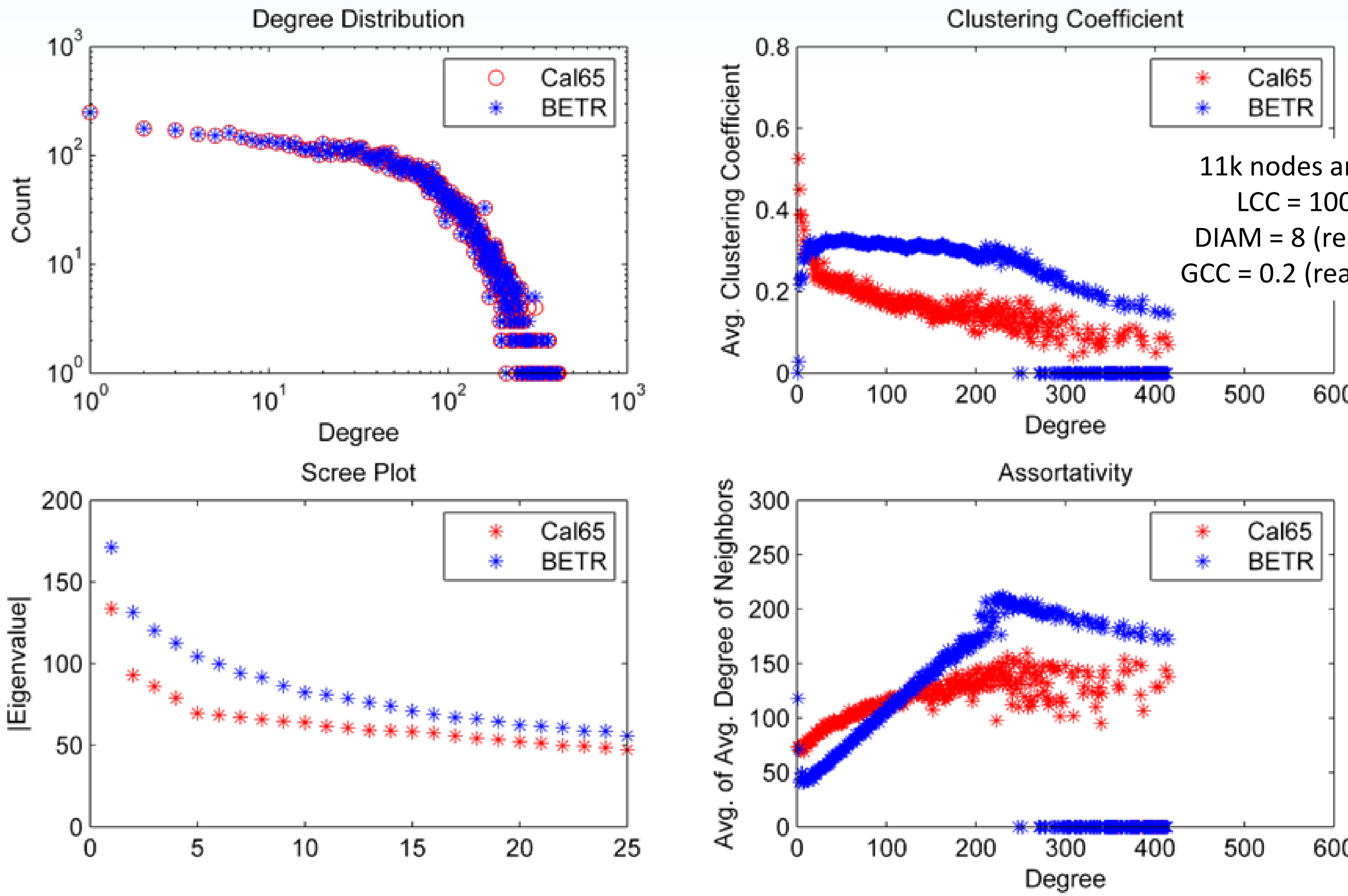
BTER
ρ = 0.6
α = 1.25



3.5k nodes and 303k edges
LCC = 100% for both
DIAM = 8 (real) vs. 6 (BTER)
GCC = 0.2 (real) vs. 0.3 (BTER)

Comparison on Social Network

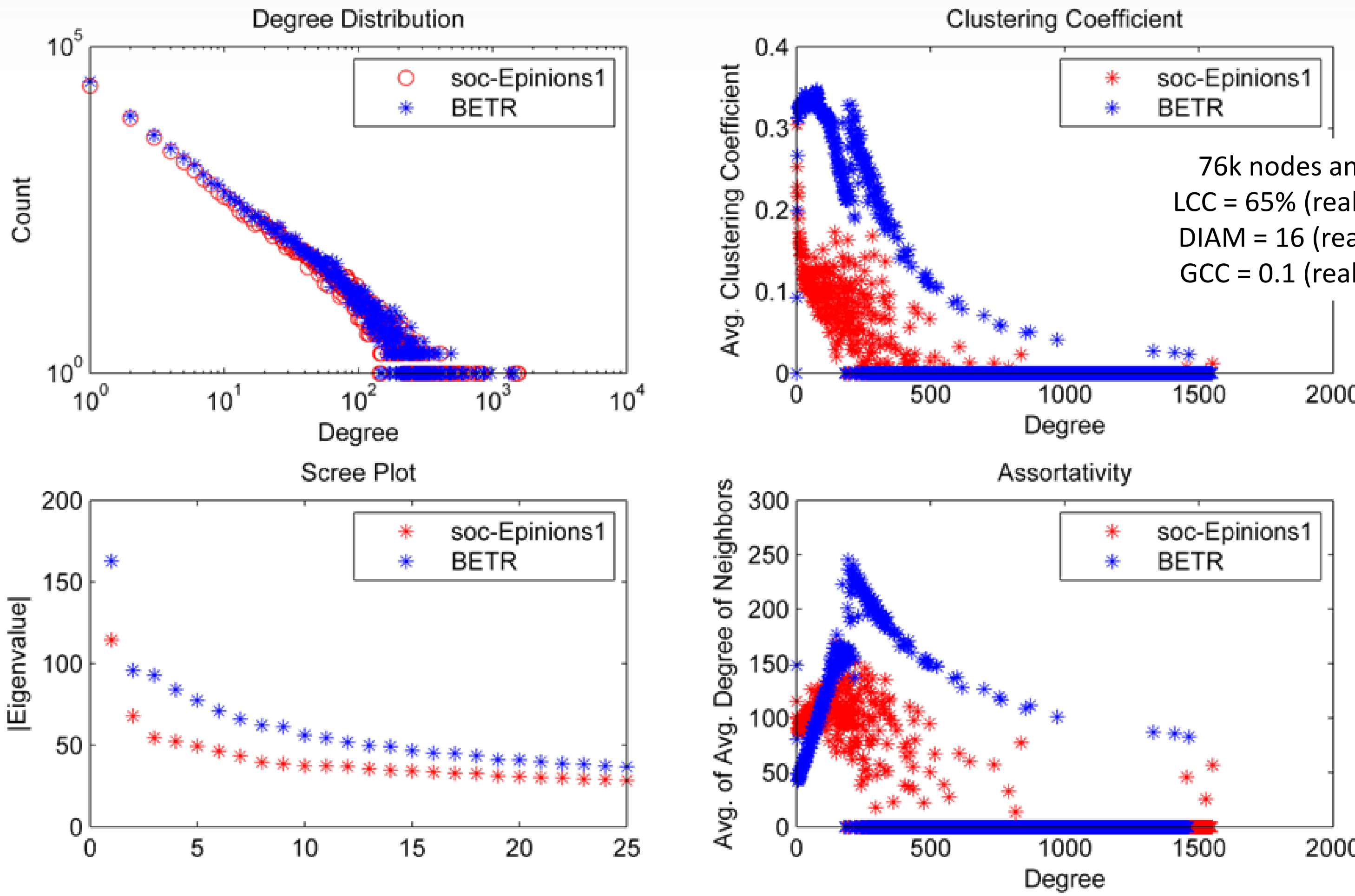
BTER
ρ = 0.6
α = 1.25



11k nodes and 703k edges
LCC = 100% for both
DIAM = 8 (real) vs. 6 (BTER)
GCC = 0.2 (real) vs. 0.3 (BTER)

Comparison for SNAP Data

BTER
ρ = 0.6
α = 1.25

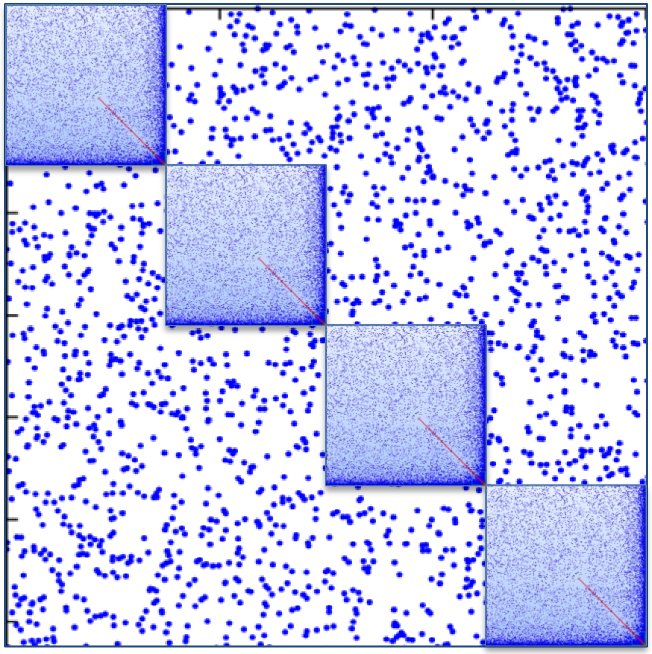


76k nodes and 442k edges
LCC = 65% (real) vs 91% (BTER)
DIAM = 16 (real) vs. 18 (BTER)
GCC = 0.1 (real) vs. 0.2 (BTER)

CONCLUSIONS AND FUTURE WORK

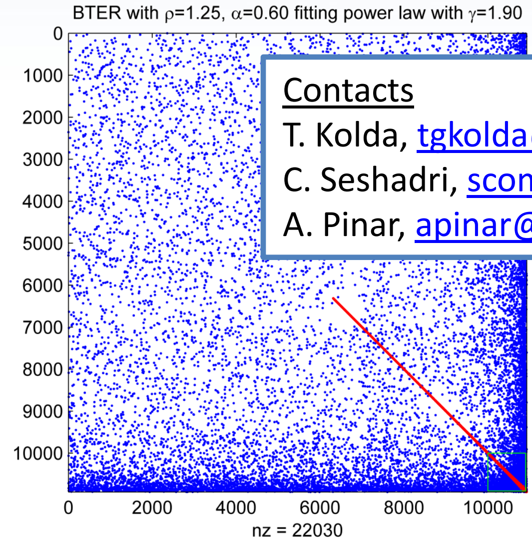
Scaling for Large Simulations

- Phase 1 is easily parallelized
  - Assign every p^{th} node to processor p
- Phase 2 requires one data exchange
  - Each processor exchanges "half-edges" with the other processors
    - Smaller-scale exchange at the price of a higher diameter
  - Can avoid the exchange altogether and instead do a match based on expectations
    - Lower accuracy in matching the degree distribution
- Hadoop MapReduce implementation coming soon

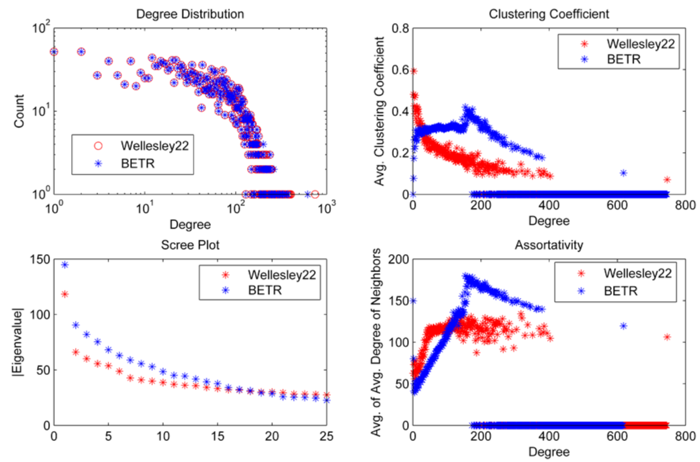


Conclusions and Future Work

- BTER meets all of our desired criteria
  - Match a variety of degree distributions
  - Community structure, as evidenced by high clustering coefficient
  - Large connected component of small diameter
  - Scalable to large problems (not yet verified)
- Future Work
  - Parallel implementations
    - MapReduce (data exchange is just one pass)
    - MPI (size of data exchange matters more in this case)
  - Theoretical underpinnings
    - Block size distribution
    - Clustering coefficients
    - Eigenvalues
  - Investigate tuning of ρ and α
    - Vary ρ and α with the degree of the clique
    - Tuning block sizes, block membership, and parameters to real data
  - Propose BTER as a candidate for Graph 500



Contacts
T. Kolda, tgkolda@sandia.gov
C. Seshadri, scomand@sandia.gov
A. Pinar, apinar@sandia.gov



EXTRA SLIDES

Erdös-Rényi (ER) Graphs

Unweighted

- Given: Fixed edge probability, ρ
- Version 1: PROB_DENSE
  - Flip independent ρ-coin for each edge
- Version 2: PROB_SPARSE
  - Pick two vertices uniformly at random to create an edge
  - Create ρN^2 edges
  - Omit duplicates & self-edges
- Version 3: DEGREE_MATCH
  - Assign every edge a degree of floor(ρN) or ceil(ρN) so that total edges = ρN^2
  - Create half-edges for all nodes
  - Randomly match
  - Remove duplicates & self-edges and repeat until stuck

Weighted (Configuration Model)

- Given: Degree distribution, **d.** M = sum(**d**) = # edges.
- Version 1: PROB_DENSE
  - Flip independent coin for each edge according to $p_{ij} = d_i d_j / M$
- Version 2: PROB_SPARSE
  - Pick two vertices according to $p_i = d_i / M$
  - Create M edges
  - Omit duplicates & self-edges
- Version 3: DEGREE_MATCH
  - Create half-edges for all nodes
  - Randomly match
  - Remove duplicates & self-edges and repeat until stuck

Outline

- Some motivations for graph models, highlighting those that matter to us
- Our 3 main goals
- Limitations of current graph models
- A note on “ER” graphs
- Our model – general description, SPY plots, block size distribution, etc.
- Our model vs WER
- Our model vs R-MAT
- Theory: # blocks, cc, diameter
- Scaling up
- Examples with scaling??
- Conclusions

Limitations of Current Models

- Configuration Models [CITE]
- Exponential Random Graphs [CITE]
- Multifactal Graph Generator [Palla, Lovász, Vicsek, PNAS 2010]
  – Not scalable (MC to match degree or CC distribution)
- Stochastic Kronecker Graphs [CITE]
  – Scalable!
  – Limited to lognormal degree distribution (with noise)
  – Very small clustering coefficients