Large scale computation of the trace of a matrix function

Giuseppe Rodriguez

Department of Mathematics and Computer Science, University of Cagliari, Italy

International Workshop on
Applied Mathematics & Quantum Information
Cagliari, November 3–4, 2016

Graphs

Let us consider a graph $G = (V, E)$, where:

- V is a set of nodes that we denote by $\{1, \ldots, n\}$;
- E is a set of edges (or arcs) between nodes.

If the arcs are oriented we say that the graph is directed (digraph), undirected otherwise. A graph is (strongly) connected if for any pairs of nodes there is an (oriented) path connecting them.

To each unweighted graph corresponds an adjacency matrix

$$A_{ij} = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ are connected,} \\ 0, & \text{if nodes } i \text{ and } j \text{ are not connected.} \end{cases}$$

If the graph is undirected the adjacency matrix is symmetric.

Complex networks

A complex network is a graph with particular properties, which is used to model interaction between entities in various fields, e.g., computer science, sociology, economics, genetics, epidemiology, etc.

Though there is often little randomness in the phenomena being studied, complex networks are generally seen as random graphs with some additional features:

- strong clustering,
- small world effect,
- scale-free structure (power law degree distribution).

Clustering index

The friend of your friend is likely to be your friend.

Global clustering is measured by the average of the clustering of the nodes

$$\text{clust}(G) = \frac{1}{n}\sum_{i=1}^{n} C(i)$$

where $C(i)$ is the probability that two nodes connected to the same node are connected themselves

$$C(i) = \frac{t_i}{T_i} = \frac{(A + A^T)^3_{ii}}{2[\text{deg\_tot}(i)\,(\text{deg\_tot}(i) - 2[A^2]_{ii})]},$$

i.e., ratio between existing and possible triangles.

Small world effect

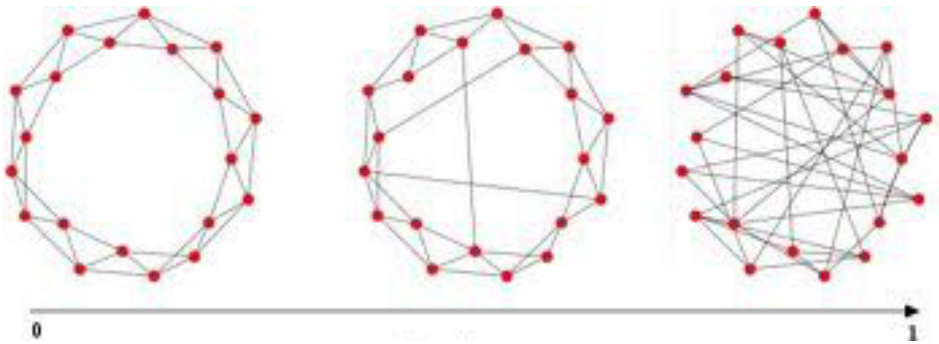
Milgram's six degree of separation.

In a *small world* network the mean distance among nodes is *small*:

$$\ell(G) := \frac{1}{N} \sum_{(i,j)\in\mathcal{C}} \text{dist}(i,j) \lessapprox \log n,$$

where $\mathcal{C} = \{(i,j) : \text{dist}(i,j) < \infty\}$.

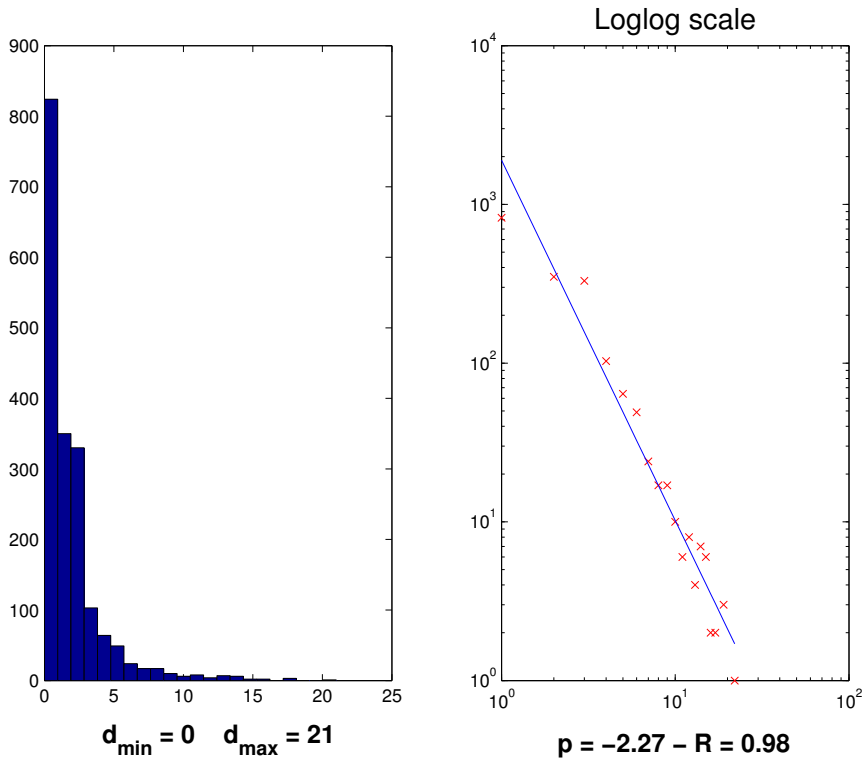
Example from [Watts and Strogatz, 1998]:



Power law degree distribution

The rich gets richer.

In a *scale-free* network the distributions of the degrees of its nodes follows a power law: $F(k) \sim ck^{-p}$, with $p > 0$.
This implies the presence of hubs, i.e., super-connected nodes.



Ranking nodes: the degree

Various indices (or metrics) have been introduced to characterize the importance of a node in terms of connection with the rest of the network.

The simplest is the degree. If the network is undirected:

$$\deg(i) = \sum_{\substack{j=1 \\ j \neq i}}^{n} A_{ij} = \sum_{\substack{j=1 \\ j \neq i}}^{n} A_{ji};$$

if the network is directed:

$$\deg_{\text{out}}(i) = \sum_{\substack{j=1 \\ j \neq i}}^{n} A_{ij} \qquad \text{and} \qquad \deg_{\text{in}}(i) = \sum_{\substack{j=1 \\ j \neq i}}^{n} A_{ji}.$$

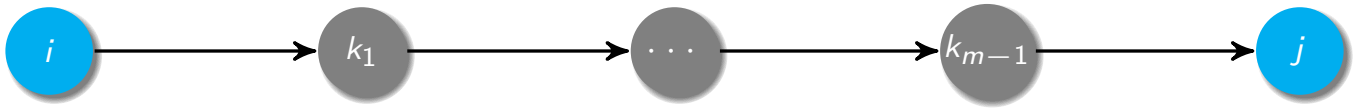
Ranking nodes: f-centrality indices

Various indices (or metrics) have been introduced to characterize the importance of a node in terms of connectivity.

A class of indices has been defined starting from matrix functions

$$f(A) = \sum_{m=0}^{\infty} c_m A^m, \qquad c_m \geq 0.$$

The key point is that $[A^m]_{ij}$ gives the number of walks of length m starting from the node i and ending at node j



Then, $[f(A)]_{ij}$ is a weighted average of all the walks connecting i to j, and describes the ease of travelling between them.

f-centrality and f-communicability

Two common choices for the coefficients in $f(A) = \sum c_m A^m$ are

$$\begin{aligned} c_m &= 1/m! \quad &\Rightarrow \quad f(A) &= \mathrm{e}^A, \\ c_m &= \gamma^m \quad &\Rightarrow \quad f(A) &= (I - \gamma A)^{-1}. \end{aligned}$$

- $[f(A)]_{ii} = \mathbf{e}_i^T f(A)\, \mathbf{e}_i$ is the f-centrality of node i,
- $[f(A)]_{ij} = \mathbf{e}_i^T f(A)\, \mathbf{e}_j$ is the f-communicability between i and j.

[Estrada, Rodríguez-Velázquez 2005, 2006], [Benzi, Boito 2010], [Estrada, Higham 2010], [Estrada, Hatano, Benzi 2012], ...

If A is not large and symmetric, we can easily compute these functions (e.g., spectral factorization, Padé approx.), but the computation is not trivial for large scale problems.

Other indices

The function $f(A)$ plays a role in the definition of other indices for the nodes of undirected or directed networks:
Katz centrality, *communicability betweenness*, *average comm.*, *total subgraph comm.*, *starting/ending convenience* [Katz 1953], [Estrada, Higham, Hatano 2009], [Estrada, Hatano, Benzi 2012], [Benzi, Klymko 2013], [Fenu, Martin, Reichel, R 2013].

These indices depend upon the row/column sums of $f(A)$, i.e.,

$$\sum_{j=1}^{n}[f(A)]_{ij} = \mathbf{e}_i^T f(A)\,\mathbf{e}, \qquad \sum_{i=1}^{n}[f(A)]_{ij} = \mathbf{e}^T f(A)\,\mathbf{e}_j,$$

which give information about walks joining node i to other nodes and walks joining other nodes to node j, respectively.

Ranking problem

To extract information from $f(A)$ the following quadratic forms may be computed

- $\mathbf{e}_i^T f(A)\,\mathbf{e}_j \rightarrow [f(A)]_{ij}$,
- $\mathbf{e}_i^T f(A)\,\mathbf{e} \rightarrow$ row-sum,

with $\mathbf{e}_i = (0, \ldots, 1, 0, \ldots 0)^T$ and $\mathbf{e} = (1, \ldots, 1)^T$.

It is often of interest to identify a subset of nodes for which a chosen index is largest (or smallest). Examples:

- find m nodes s.t. $\max_i \mathbf{e}_i^T f(A)\mathbf{e}_i$ (f-subgraph centrality);
- find m nodes s.t. $\max_i \mathbf{e}_i^T f(A)\mathbf{e}$ (f-starting convenience).

[Fenu, Martin, Reichel, R, SISC 2013],
[Baglama, Fenu, Reichel, R, LAA 2013],

Approximation by Gauss quadrature

The actual value of $\mathbf{u}^T f(A)\,\mathbf{v}$ ($A = A^T$, $\mathbf{u}, \mathbf{v} = \mathbf{e}_i, \mathbf{e}_j, \mathbf{e}, \dots$) can be obtained by Gauss quadrature [Golub, Meurant 1993...2010]. This approach was proposed for subgraph centralities and communicabilities in [Benzi, Boito 2010].

Indeed, under suitable assumptions on f, starting from

$$\mathbf{u}^T f(A)\,\mathbf{u} = \sum_{i=1}^{n} f(\lambda_i)\omega_i^2 = \int f(t)d\omega(t),$$

the connection between the symmetric Lanczos process and orthogonal polynomials leads to bound the quadratic form by sequences of Gauss and Gauss–Radau quadrature formulas

$$\mathcal{G}_{k-1} f < \mathcal{G}_k f < \mathbf{u}^T f(A)\,\mathbf{u} < \mathcal{R}_{k+1} f < \mathcal{R}_k f.$$

Complex networks and the *trace* operator

The Estrada index of a graph, defined as

$$\text{trace}(\exp(A)) = \sum_{i=1}^{n} [\exp(A)]_{ii} = \sum_{i=1}^{n} \exp(\lambda_i),$$

provides a global characterization of the graph; see [Estrada, Rodríguez-Velázquez 2005], [Estrada, Hatano, Benzi 2012].

The clustering index of a network depends upon the number T of existing triangles, which can be obtained by computing

$$T = \frac{1}{6} \text{trace}(A^3).$$

Large scale computation of trace($f(A)$)

Our approach is the following

$$\text{trace}(f(A)) = \sum_{j=1}^{\widetilde{n}} \text{trace}(E_j^T f(A) E_j).$$

where

$$E_j = [\mathbf{e}_{k(j-1)+1}, \ldots, \mathbf{e}_{\min\{jk,m\}}],$$

for $j = 1, 2, \ldots, \widetilde{n} = \left\lfloor \frac{m+k-1}{k} \right\rfloor$.

Then, it is essential to approximate at a prescribed accuracy

$$\text{trace}(W^T f(A) W),$$

for $W \in \mathbb{R}^{m \times k}$, efficiently and without actually computing $f(A)$.

The global Lanczos method

For $W_1, W_2 \in \mathbb{R}^{n\times k}$, we define the inner product and induced norm

$$\langle W_1, W_2\rangle := \text{trace}(W_1^T W_2), \qquad \|W_1\|_F := \langle W_1, W_1\rangle^{1/2}.$$

For $A = A^T$ it holds

$$A[V_1, \ldots, V_\ell] = [V_1, \ldots, V_\ell]\,\widehat{T}_\ell + \beta_{\ell+1} V_{\ell+1} E_\ell^T,$$

where $V_j \in \mathbb{R}^{n\times k}$, $\langle V_i, V_j\rangle = \delta_{ij}$, $\widehat{T}_\ell = T_\ell \otimes I_k$, and

$$T_\ell = \begin{bmatrix} \alpha_1 & \beta_2 & & & \mathsf{O} \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \cdot & \cdot & \\ & & \cdot & \cdot & \beta_\ell \\ \mathsf{O} & & & \beta_\ell & \alpha_\ell \end{bmatrix}.$$

[Jbilou, Messaoudi, Sadok 1999]
[Elbouyahyaoui, Messaoudi, Sadok 2009]

Global Gauss quadrature rules

In [Bellalij, Reichel, R, Sadok, ApNuM 2015] we considered the expression

$$\begin{aligned} \mathcal{I}f &:= \text{trace}(W^T f(A) W) && V_1 = W/\|W\|_F \\ &= \|W\|_F^2 \,\text{trace}(V_1^T f(A) V_1), && A = U\Lambda U^T \\ &= \|W\|_F^2 \,\text{trace}(\widetilde{V}_1^T f(\Lambda) \widetilde{V}_1) && \widetilde{V}_1 = U^T V_1 \end{aligned}$$

For any $i = 1, 2, \ldots, k$ we can write

$$\mathbf{e}_i^T \widetilde{V}_1^T f(\Lambda) \widetilde{V}_1 \mathbf{e}_i = \sum_{j=1}^{m} f(\lambda_j)\mu_j^2 = \int f(\lambda) d\mu_i(\lambda),$$

from which, letting $\mu(\lambda) := \sum_{i=1}^{k} \mu_i(\lambda)$, it follows

$$\mathcal{I}f = \|W\|_F^2 \sum_{i=1}^{k} \int f(\lambda) d\mu_i(\lambda) = \|W\|_F^2 \int f(\lambda) d\mu(\lambda)$$

Global Gauss quadrature rules

The global Lanczos method gives $V_j = p_{j-1}(A)V_1$, so that

$$\delta_{ij} = \langle V_j, V_i \rangle = \langle p_{j-1}(A)V_1, p_{i-1}(A)V_1 \rangle = \int p_{j-1}(\lambda)p_{i-1}(\lambda)d\mu(\lambda)$$

implies that

$$\mathcal{G}_\ell f = \|W\|_F^2 \mathbf{e}_1^T f(T_\ell)\mathbf{e}_1 \quad \text{is a Gauss quadrature rule}$$

which is exact for polynomials of degree $2\ell - 1$.

Quick proof: $p_j(\lambda)$ OPs recursion coefficients $\rightarrow T_\ell = QDQ^T$.

Global Gauss quadrature rules

By extending T_ℓ to $T_{\ell+1,\zeta}$, with an additional eigenvalue at ζ, we obtain that

$\mathcal{R}_{\ell+1,\zeta} f = \|W\|_F^2 \mathbf{e}_1^T f(T_{\ell+1,\zeta})\mathbf{e}_1$ is a Gauss–Radau quadrature rule

which is exact for polynomials of degree 2ℓ.

If the derivatives of f behave nicely and $\zeta \geq \max_i \lambda_i$, we obtain a sequence of lower and upper bounds

$$\mathcal{G}_{\ell-1} f < \mathcal{G}_\ell f < \mathcal{I} f < \mathcal{R}_{\ell+1,\zeta} f < \mathcal{R}_{\ell,\zeta} f.$$

We stop the iteration when

$$\frac{|\mathcal{R}_{\ell+1,\zeta} f - \mathcal{G}_\ell f|}{2|\mathcal{G}_\ell f|} < \tau.$$

Results for the block algorithm

Matrix	Nodes	Global Lanczos time	$k_{\min}$	Scalar Lanczos time	SF	`expm` time	SF
Email	1133	3.5e-01	80	3.5e+00	(10)	1.2e+01	(34)
Yeast	2114	4.7e-01	60	3.2e+00	(7)	1.0e+01	(21)
Power	4941	1.9e+00	40	1.3e+01	(7)	2.1e+01	(11)
Internet	22963	1.2e+02	8	3.3e+02	(3)	-	-
Collab.	40421	4.8e+02	40	1.3e+03	(3)	-	-
Facebook	63731	2.6e+03	60	8.8e+03	(3)	-	-

Execution times for computing the Estrada index.
SF is the speedup factors with respect to the block method,
i.e., the ratio between the computing times.

Parallel computation of the Estrada index

The expression

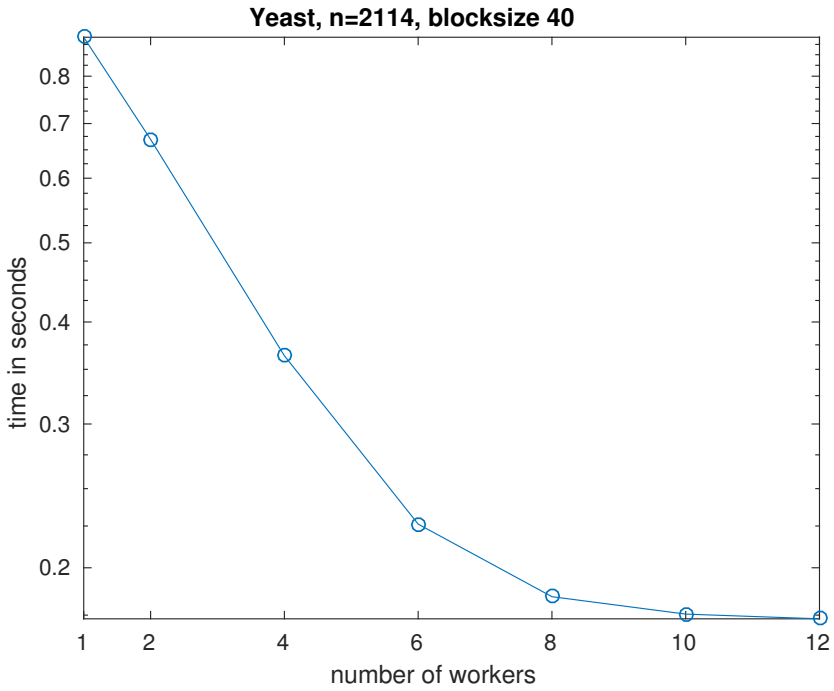
$$\text{trace}(\exp(A)) = \sum_{j=1}^{\widetilde{n}} \text{trace}(E_j^T \exp(A) E_j).$$

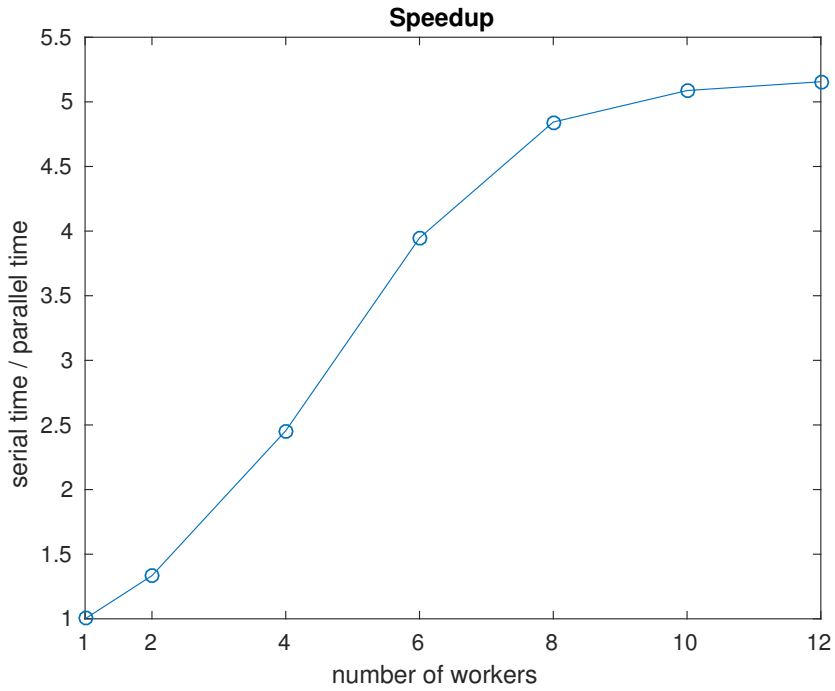
allows two levels of parallelization:

1. implicit parallelization triggered by the use of global Lanczos to compute $\text{trace}(E_j^T \exp(A) E_j)$;
2. explicit parallelization obtained by computing each term of the summation on a different *worker*.

E. Cannas and A. Concas investigated this aspect while working on a bachelor thesis, using the Parallel Computing Toolbox of Matlab.

Results for the parallel algorithm

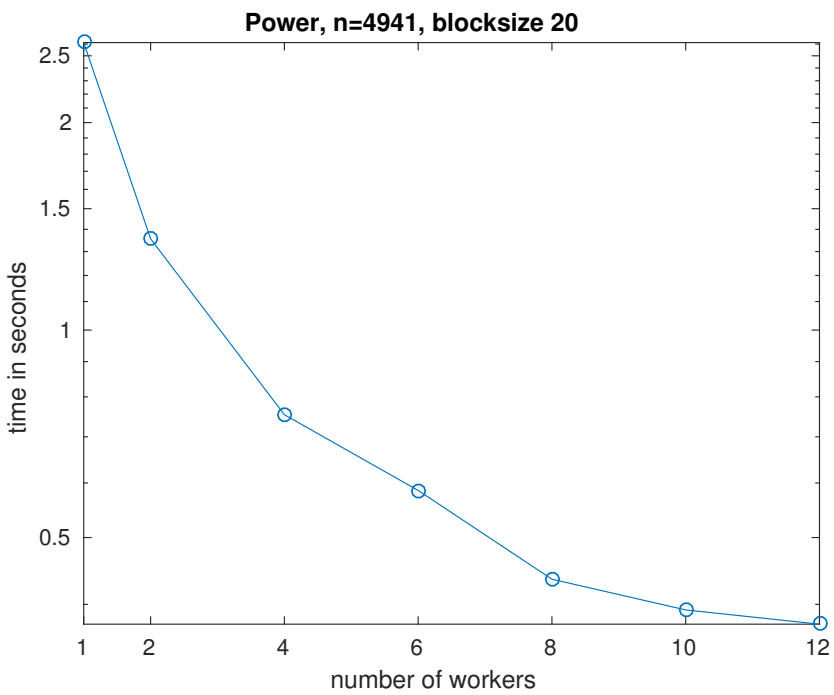


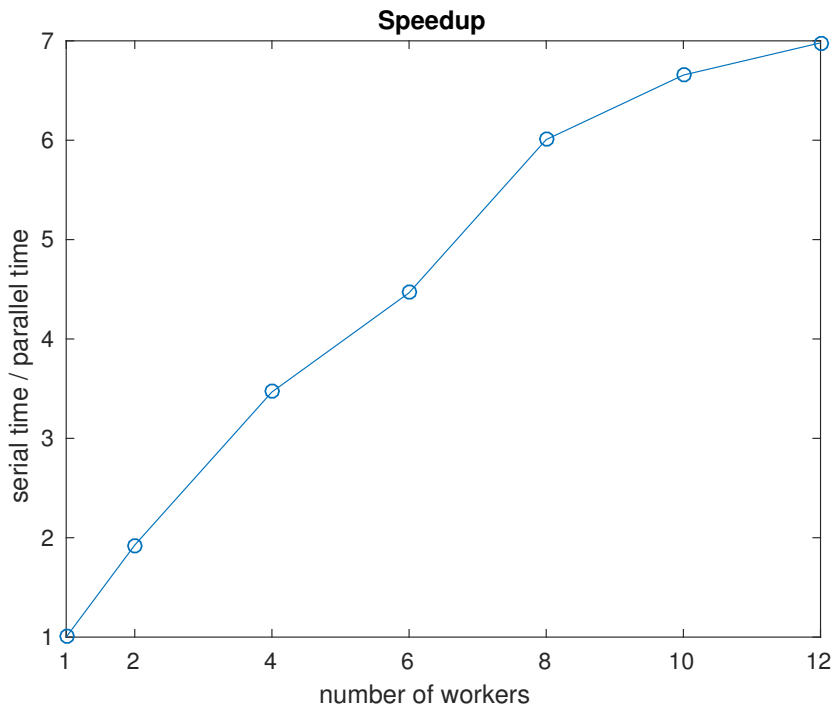


Computation time and speedup for the yeast network
(2114 nodes, 4480 arcs)

Intel Xeon E5-2620, 2 CPUs, 12 cores
Matlab 9 - Linux Debian Stretch

Results for the parallel algorithm

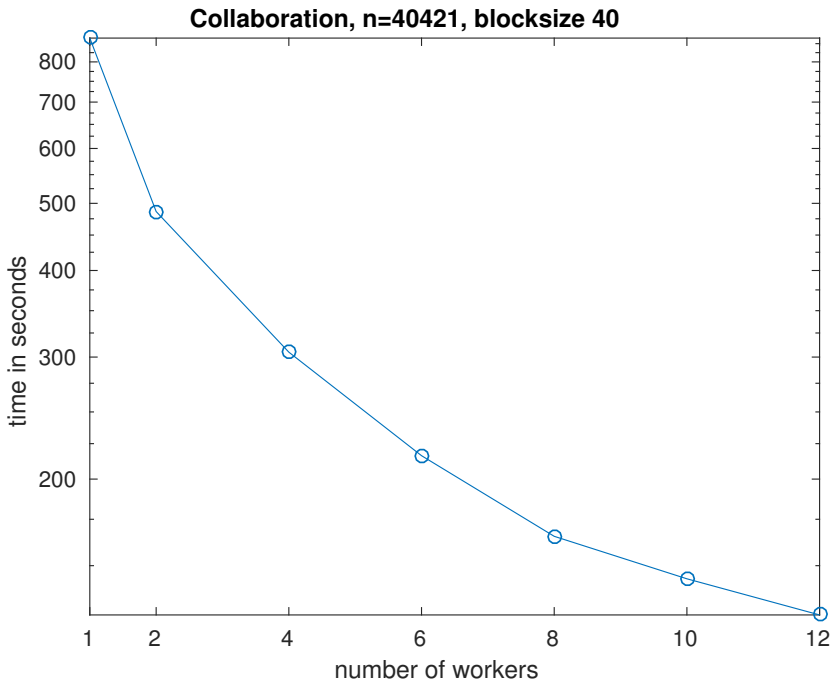


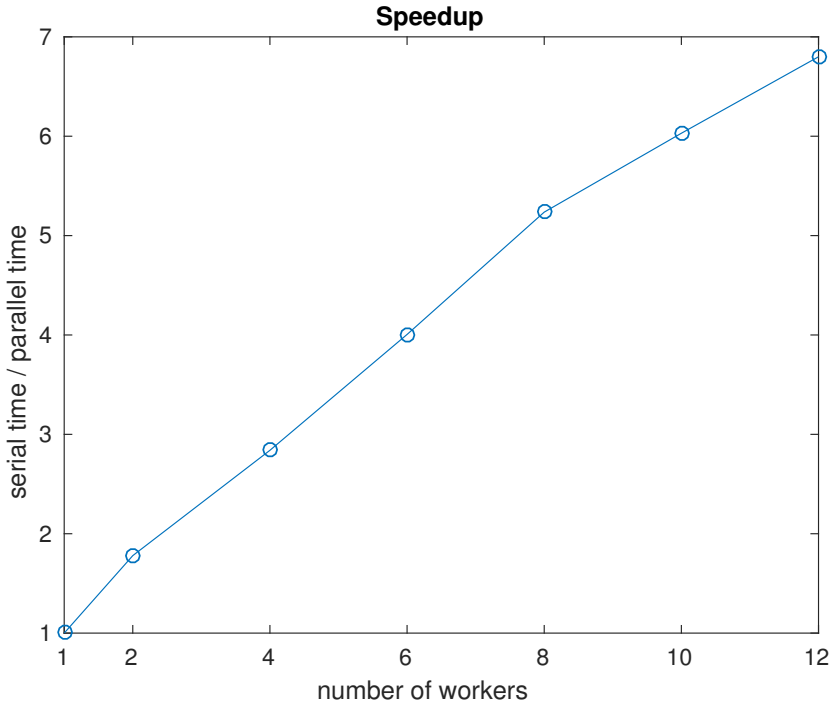


Computation time and speedup for the Power network
(4941 nodes, 13188 arcs)

Intel Xeon E5-2620, 2 CPUs, 12 cores
Matlab 9 - Linux Debian Stretch

Results for the parallel algorithm





Computation time and speedup for the Collaboration network (40421 nodes, 351384 arcs)

Intel Xeon E5-2620, 2 CPUs, 12 cores
Matlab 9 - Linux Debian Stretch

Thanks for your attention!