Using distributed local information to improve global performance in Grids

P. Verghelet, D. Fernández Slezak, P. Turjanski, E. Mocskos

Laboratorio de Sistemas Complejos
Departamento de Computación
FCEN - UBA

Information Resources in Grid Computing

- Public collaboration across many countries and networks, with geographically distributed and heterogeneous resources
- Optimizing requests requires **up-to-date information** about the grid
- Dynamics of resources information cannot be captured using a **static hierarchy (defacto implementation)** in medium-to-large enviroments.
- New policies needed. Growing interest in the interaction of Grid Computing and the **P2P paradigm**, pushing towards **automated and scalable** solutions.

Objectives

- Present a new Best-Neighbor policy: fBN.
- We study peformance compared to previous Best-Neighbor implementation and other policies: Random, Hierarchical and Super-Peer policies.

Best-Neighbor policy

- No human administration at all.
- Each node collects information from the network and next node to query is selected using accumulated information.
- The method consists of two stages:
  1. *learning stage*: without information, the next node is randomly selected.
  2. *working stage*: query to the best-neighbor node.

Best-Neighbor strongly depends on knowing as much nodes as possible in the network.

Evaluation of system performance

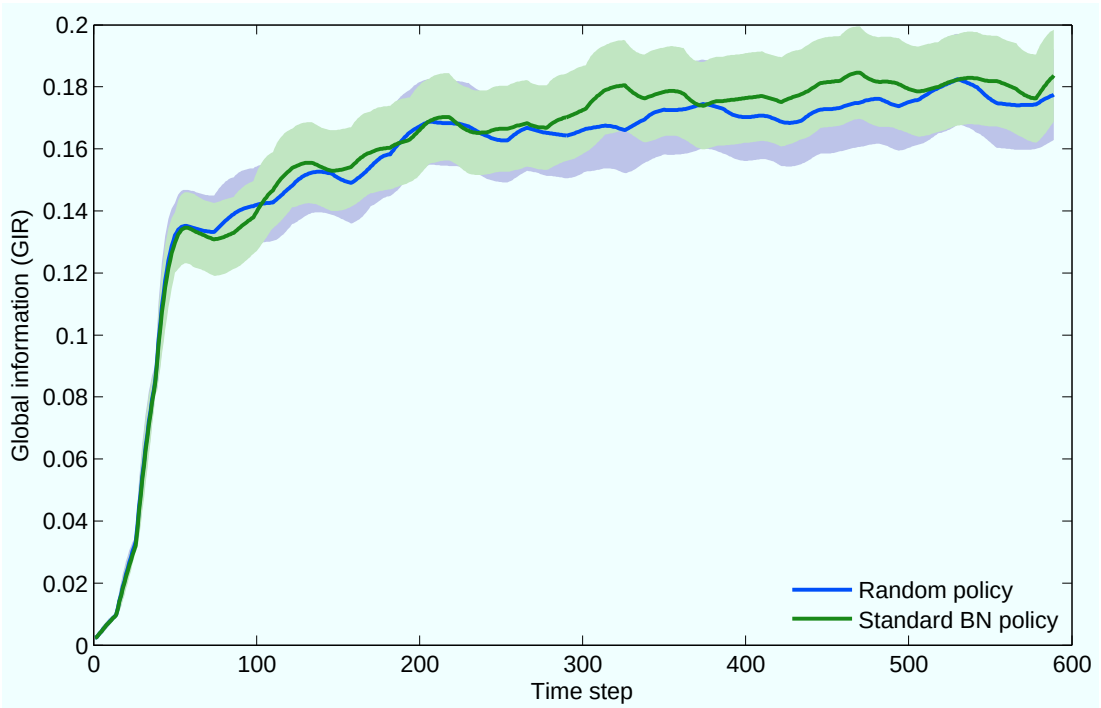
LIR: captures the amount of information that a particular host has from all the entire grid in a single moment. For the host k, LIR_k is:

$$LIR_k = \frac{\sum_{h=1}^{N} f(age_h, expiration_h) \cdot resourceCount_h}{totalResourceCount}$$

GIR: captures the amount of information that the whole grid knows of itself, calculated as the mean value of every node's LIR.

Standard BN equals Random

- As system scales up, Best-Neighbor behaves similar to Random policy.



Evolution of GIR for Random and BN, in 400-nodes exponential network.

Two possible causes...

We analyze the factors that may explain these results:

(1) Amount of known nodes from the network The learning stage duration is directly related to the amount of known nodes. As faster the nodes are discovered, the sooner this stage finishes.

(2) How to select the best node to query BN uses a *scoring function* for ranking nodes. This function may be tuned for maximum performance.

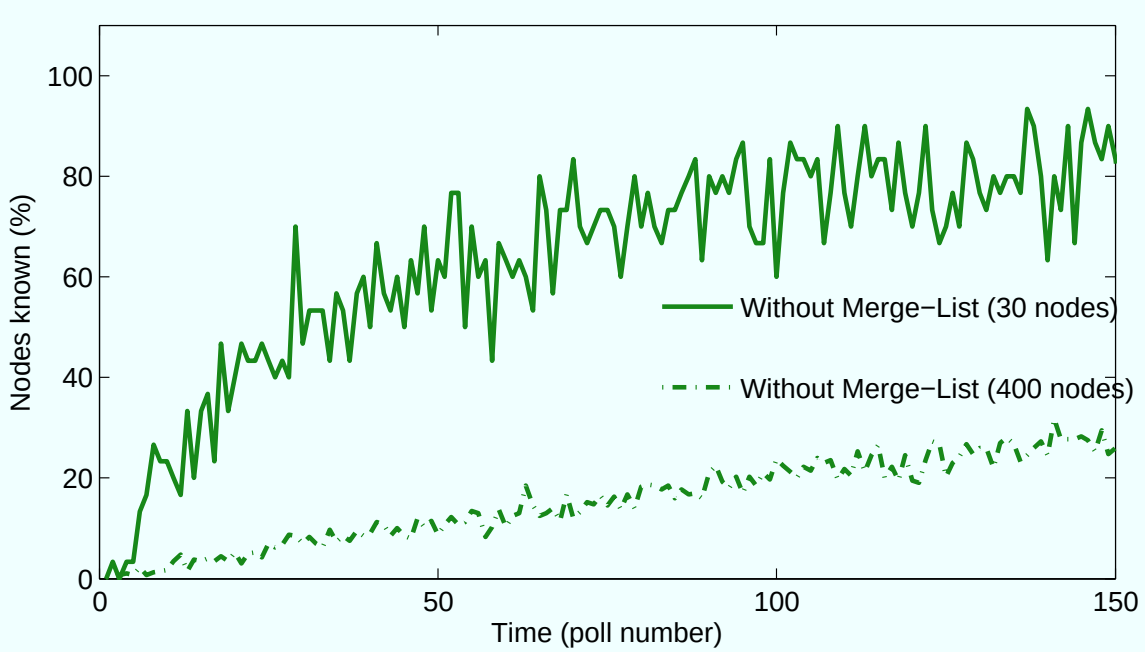
Standard BN Scoring function

$$f_{scoring} = a * \text{RES\_COUNT} - b * \text{RTT} - c * \text{RESPONSE\_FAILED}$$

where RES_COUNT is the amount of available resources in the node, RTT is the Roud Trip Time and RESPONSE_FAILED counts the number of message looses.

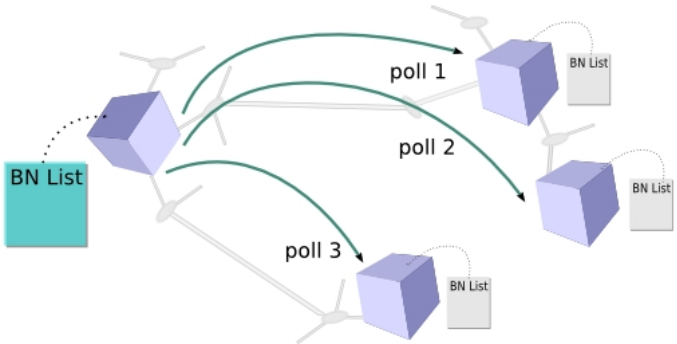
(1) Learning stage takes too long

- BN learning stage uses Random policy.
- Learning stage stays active too long due to the slow speed of node discovery.
- As systems scales up, this situation worsens

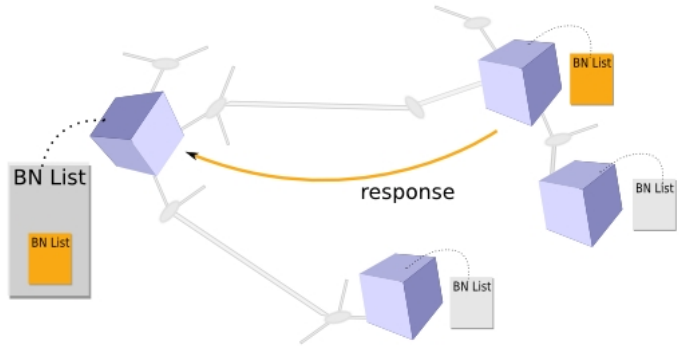


(1) Merging lists

- To tackle this problem, we choose *merging lists* as the solution.



In blue the best-neighbors list



On response, *merging lists*

Merging List

This technique consists of sharing the list of neighbors that a particular node has to any other node whenever the node responses to a query.

(2) Adding the amount of own resources

- We incorporate a new term which captures information about the amount of local resources of the node.

New scoring function

Standard: $f_{scoring} = a * \text{RES\_COUNT} - b * \text{RTT} - c * \text{RESPONSE\_FAILED}$

New: $f_{new} = f_{scoring} + d * \textbf{OWN\_RES\_COUNT}$

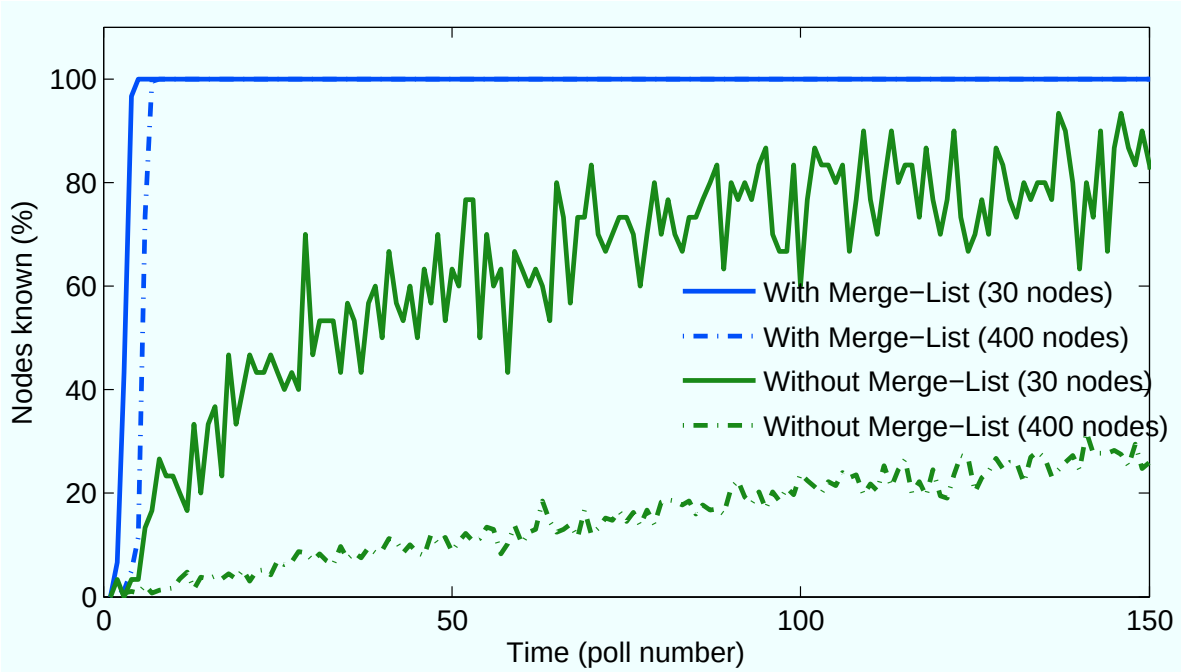
where OWN_RES_COUNT is the amount of local resources in the node.

fBN policy

We present the fBN policy, an implementation of Best-Neighbor that incorporates both mentioned improvements:

1. Merging Lists
2. Improved scoring function

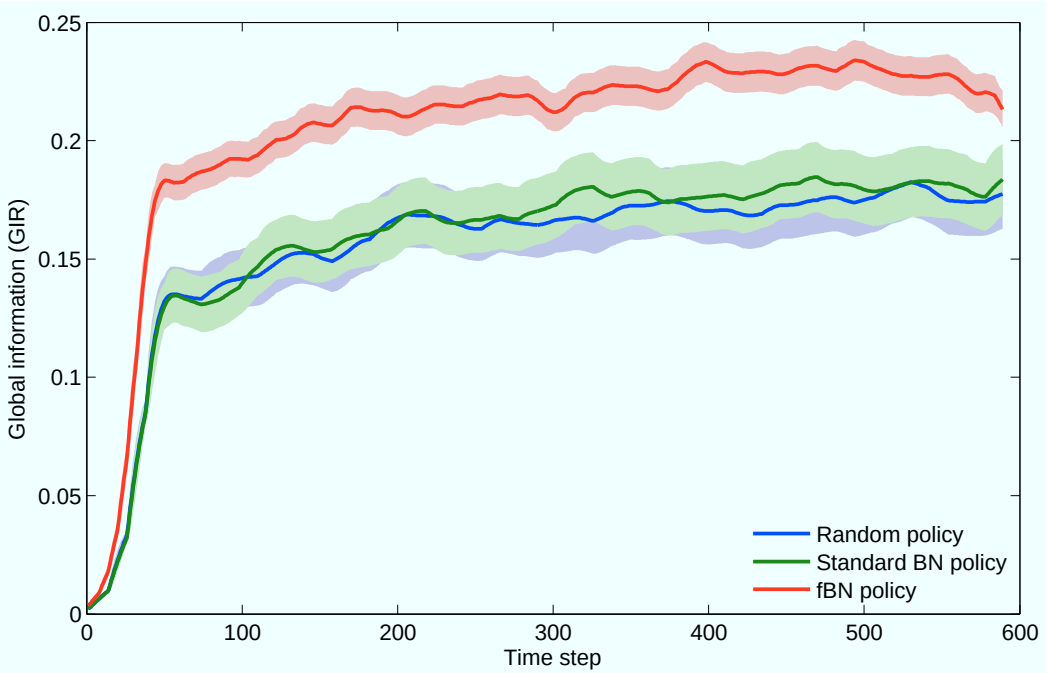
fBN policy



Merging Lists

Using this improvement, all the nodes on the network are quickly known and the working stage can initiate shortly(blue lines).

fBN policy



400-nodes exponential network.

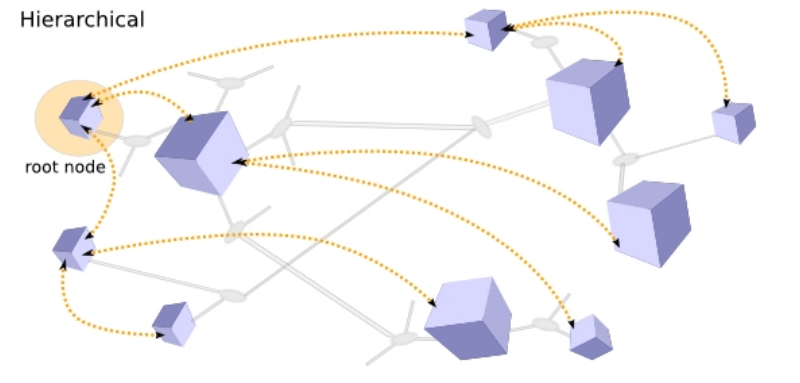
New scoring function

Evolution of GIR for Random, BN and fBN.
fBN outperforms both others policies (red line)

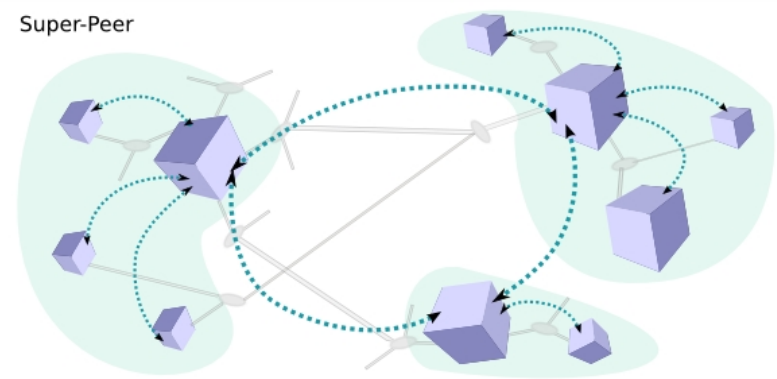
Other policies

What about the others policies?

This encouraging results suggests to us compare fBN policy with the others mentioned policies: Hierachical and Super-Peer



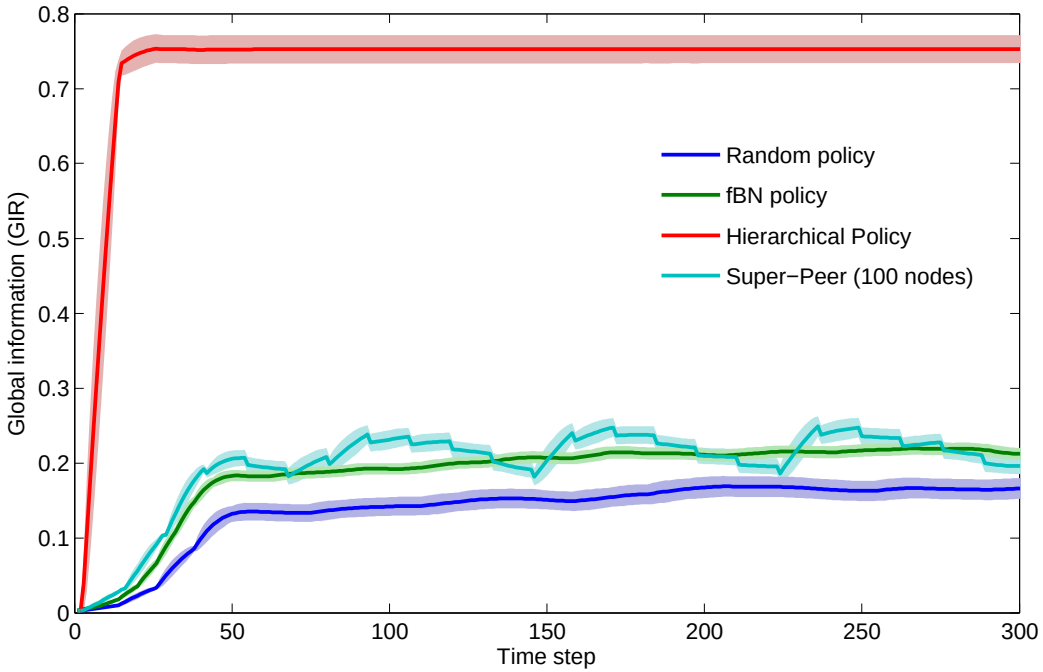
Hierarchical policy overlay (orange dot lines)



Super-Peer policy overlay (blue dot lines)

Other policies

Evolution of GIR for fBN, Super-Peer, Random and Hierarchical.
400-nodes exponential network.



Comparission with other policies

fBN shows similar behavior to Super-Peer: unsupervised method may obtain comparable result to supervised ones. Both results perform better Random, but very far from the hierarchical policy.

Conclusions

Two modifications are introduced to improve the performance of Best-Neighbor policy obtaining fBN:

1. merge the lists of neighbors during the learning stage to decrease the length of this phase.
2. a new term which considers the amount of local resources provided by the node is added to the scoring function.

- fBN presents a shorter learning phase, not affected by the system size increment.
- fBN outperforms Random policy and shows similar behavior to Super-peer.
- Hierarchical shows the best performance, but is the policy needing more setup and administration.
- fBN results in a good trade-off between fully automated policy and obtained performance.

Thank you !!