On the complexity of Policy Iteration for PageRank Optimization

Romain Hollanders

Joint work with Raphaël Jungers and Jean-Charles Delvenne

Université catholique de Louvain

June 2011
PageRank is the average time-portion spent in a node during an infinite random walk.
PageRank is the average time-portion spent in a node during an infinite random walk.
PageRank is the average time-portion spent in a node during an infinite random walk.
PageRank Optimization by edge selection

A not so trivial task...
Several algorithms have been proposed
But which one should we use?

1. **Original approach**: polynomial time
   - But only approximates the optimal solution
   - (Ishii & Tempo, 2008)
Several algorithms have been proposed
But which one should we use?

1. **Original approach**: polynomial time
   But only approximates the optimal solution
   (Ishii & Tempo, 2008)

2. **Linear Programming**: exact, polynomial time
   But does not take the full problem’s specificity into account
   (CSáji, Jungers & Blondel, 2009)
Several algorithms have been proposed
But which one should we use?

1. **Original approach**: polynomial time
   But only approximates the optimal solution
   (Ishii & Tempo, 2008)

2. **Linear Programming**: exact, polynomial time
   But does not take the full problem’s specificity into account
   (CSáji, Jungers & Blondel, 2009)

3. **Algorithm based on Policy Iteration**: exact, very efficient in practice
   But few bounds on its theoretical complexity
Several algorithms have been proposed
But which one should we use?

1. **Original approach**: polynomial time  
   But only approximates the optimal solution  
   (Ishii & Tempo, 2008)

2. **Linear Programming**: exact, polynomial time  
   But does not take the full problem’s specificity into account  
   (CSáji, Jungers & Blondel, 2009)

3. **Algorithm based on Policy Iteration**: exact, very efficient in practice  
   But few bounds on its theoretical complexity

Our main focus
Outline

1. The Max-PageRank Problem
   Which problem do we want to solve?

2. The PageRank Iteration algorithm
   How do we solve the problem?

3. Results
   What did we find about the algorithm?
Which fragile edge should we activate?
To maximize the PageRank of $v$ or minimize its first hitting time
Which fragile edge should we activate?

We formulate the problem as a Stochastic Shortest Path problem.
Which fragile edge should we activate?

We formulate the problem as a Stochastic Shortest Path problem.
Which fragile edge should we activate?
To maximize the PageRank of $v$ or minimize the distance from $v_s$ to $v_t$
1 The Max-PageRank Problem
   Which problem do we want to solve?

2 The PageRank Iteration algorithm
   How do we solve the problem?

3 Results
   What did we find about the algorithm?
The PageRank Iteration algorithm

At each step, we switch all fragile edges that greedily improve the first return time of $v$.
Iteration 1: Evaluation step
Initial policy: all fragile edges are OFF

\[ S_1 = \{ \} \]
Iteration 1 : Evaluation step
Initial policy : all fragile edges are OFF

\[ S_1 = \{ \} \]
Iteration 1: Improvement step

It is good to step from a distance $d$ from $v_t$ to a distance $< d - 1$

$S_1 = \{ \}$

$T_1 = \{ A, C, D \}$
Iteration 1 : Improvement step

\[ S_1 = \{ \} \]
\[ S_2 = \{ A \ C \ D \} \]
\[ T_1 = \{ A \ C \ D \} \]
Iteration 2: Evaluation step

\[ S_1 = \{ \} \]
\[ S_2 = \{ A \ C \ D \} \]
\[ T_1 = \{ A \ C \ D \} \]
Iteration 2 : Improvement step

\[ S_1 = \{ \} \]
\[ S_2 = \{ A \ C \ D \} \]
\[ T_1 = \{ A \ C \ D \} \]
\[ T_2 = \{ A \ C \} \]
Iteration 2: Improvement step

$S_1 = \{ \}$
$S_2 = \{ A \ C \ D \}$
$S_3 = \{ D \}$

$T_1 = \{ A \ C \ D \}$
$T_2 = \{ A \ C \}$
Iteration 3: Evaluation step

\[ S_1 = \{ \} \]
\[ S_2 = \{ A, C, D \} \]
\[ S_3 = \{ D \} \]
\[ T_1 = \{ A, C, D \} \]
\[ T_2 = \{ A, C \} \]
Iteration 3 : Improvement step

No improvements available

\[ S_1 = \{ \} \]
\[ S_2 = \{ A \ C \ D \} \]
\[ S_3 = \{ D \} \]

\[ T_1 = \{ A \ C \ D \} \]
\[ T_2 = \{ A \ C \} \]
\[ T_3 = \{ \} \]

The solution is optimal! : \( K = 3 \)
Outline

1. The Max-PageRank Problem
   Which problem do we want to solve?

2. The PageRank Iteration algorithm
   How do we solve the problem?

3. Results
   What did we find about the algorithm?
In practice, PRI is by far the best method

In terms of execution time
The number of iterations of PRI seems to grow at most linearly with respect to the problem size.
But what is the worst case complexity of PRI?
How many iterations?

1. PRI takes at most $2^f$ iterations (Howard, 1960)
   Trivial since each possible policy is considered at most once
But what is the worst case complexity of PRI?
How many iterations?

1. PRI takes at most $2^f$ iterations (Howard, 1960)
   Trivial since each possible policy is considered at most once

2. PRI takes at most $O\left(\frac{2^f}{f}\right)$ iterations (Mansour & Singh, 1999)
   First non trivial bound
But what is the worst case complexity of PRI?

How many iterations?

1. PRI takes at most $2^f$ iterations (Howard, 1960)
   Trivial since each possible policy is considered at most once

2. PRI takes at most $O(2^f/f)$ iterations (Mansour & Singh, 1999)
   First non trivial bound

3. For some cases, polynomial upper bounds (Ye, 2010)
   and exponential lower bounds (Fearnley, 2010) also exist
   But they do not apply here
But what is the worst case complexity of PRI?
How many iterations?

1. PRI takes at most $2^f$ iterations (Howard, 1960)
   Trivial since each possible policy is considered at most once

2. PRI takes at most $O\left(\frac{2^f}{f}\right)$ iterations (Mansour & Singh, 1999)
   First non trivial bound

3. For some cases, **polynomial** upper bounds (Ye, 2010)
   and **exponential** lower bounds (Fearnley, 2010) also exist
   But they do not apply here

Can we do better? : Maybe!
Our tools

We define:

1. The configuration set $S_k$ ($\sim$ the policy at iteration $k$)

   $S_k$ contains all activated fragile edges from iteration $k$. 

$S_{k+1} = S_k \oplus T_k$
Our tools

We define:

1. The configuration set $S_k$ (∼ the policy at iteration $k$)
   $S_k$ contains all activated fragile edges from iteration $k$

2. The improvement set $T_k$
   Switching elements of $T_k$ will improve $S_k$
Our tools

We define:

1. The configuration set $S_k$ (the policy at iteration $k$)
   $S_k$ contains all activated fragile edges from iteration $k$

2. The improvement set $T_k$
   Switching elements of $T_k$ will improve $S_k$

In the improvement step of PRI, we update $S_k$ as follows:

$$S_{k+1} = S_k \oplus T_k$$
Strong properties hint toward a linear bound on the complexity of PRI
But something is still missing...

The following properties hold for the improvement sets:

1. \( \exists i < j \) such that \( T_i \subseteq T_j \)

2. \( \exists i < j \) such that \( T_i \oplus \cdots \oplus T_{j-1} \subseteq T_j \)

3. \( \exists i < j \) such that \( T_i \subseteq T_{i+1} \oplus \cdots \oplus T_j \)

Properties derived from Mansour & Singh (1999)
Conclusions

- PRI is an efficient algorithm for the PageRank Optimization problem
  And other problems alike
Conclusions

- PRI is an efficient algorithm for the PageRank Optimization problem and other problems alike.

- In practice, $K$ seems to grow at most linearly w.r.t. the problem size or even logarithmically?
Conclusions

- PRI is an efficient algorithm for the PageRank Optimization problem and other problems alike.

- In practice, $K$ seems to grow at most linearly w.r.t. the problem size or even logarithmically?

- In theory, $K$ could be exponential. The gap between theoretical and experimental guarantees is huge.
Conclusions

- PRI is an efficient algorithm for the PageRank Optimization problem
  And other problems alike

- In practice, $K$ seems to grow at most linearly w.r.t. the problem size
  Or even logarithmically?

- In theory, $K$ could be exponential
  The gap between theoretical and experimental guarantees is huge

We have new properties that can be used to reduce the gap and eventually solve our webmaster’s problem efficiently.
Thanks for your attention!