On the complexity of optimizing PageRank

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.
\[
\min_{\pi \in \Pi} x_1 \quad \text{such that} \quad x = P^\pi x + 1
\]
PageRank Optimization (PRO)
PRO
PageRank Optimization
PRO
PageRank Optimization

1. A set of nodes
2. Sets of fixed and free edges
3. Uniform transition probabilities
4. Unit transition costs
5. Minimize the expected first return time
PRO

Markov Decision Processes

PageRank Optimization

MDP

Policy Iteration (PI)
Linear Programming Resolution

PageRank Iteration (PRI)
Linear Programming Resolution

A set of states
A set of actions
Any transition probabilities
Any transition costs
Minimize the expected total-cost

A set of nodes
Sets of fixed and free edges
Uniform transition probabilities
Unit transition costs
Minimize the expected first return time
PRO

A set of states

A set of actions

Any transition probabilities

Any transition costs

Minimize the expected total-cost

MDP
Markov
Decision
Processes
PRO

PageRank Optimization

1. A set of nodes
2. Sets of fixed and free edges
3. Uniform transition probabilities
4. Unit transition costs
5. Minimize the expected first return time

MDP

Markov Decision Processes

1. A set of states
2. A set of actions
3. Any transition probabilities
4. Any transition costs
5. Minimize the expected total-cost

Policy Iteration (PI)

Linear Programming Resolution

PageRank Iteration (PRI)

Linear Programming Resolution
PRO

Optimization

1. A set of nodes
2. Sets of fixed and free edges
3. Uniform transition probabilities
4. Unit transition costs
5. Minimize the expected first return time

MDP

Markov Decision Processes

1. A set of states
2. A set of actions
3. Any transition probabilities
4. Any transition costs
5. Minimize the expected total-cost

Policy Iteration (PI)

Linear Programming Resolution

PageRank Iteration (PRI)

Linear Programming

Resolution

Linear Programming
PageRank Optimization

1. A set of nodes
2. Sets of fixed and free edges
3. Uniform transition probabilities
4. Unit transition costs
5. Minimize the expected first return time

MDP

1. A set of states
2. A set of actions
3. Any transition probabilities
4. Any transition costs
5. Minimize the expected total cost

Policy Iteration (PI)

Linear Programming Resolution

PageRank Iteration (PRI)

Linear Programming Resolution

Resolution

Linear Programming
PageRank Iteration: how it works

0. Initialize: Choose an initial policy $\pi_0$ (arbitrarily)
PageRank Iteration: how it works

0. Initialize: Choose an initial policy $\pi_0$ (arbitrarily)

while $\pi_k \neq \pi_{k-1}$ do

1. Evaluate $\pi_k$: compute the first return times $x^{\pi_k}$ of the nodes:

$$x^{\pi_k} = P^{\pi_k}x^{\pi_k} + 1$$
PageRank Iteration: how it works

0. Initialize: Choose an initial policy $\pi_0$ (arbitrarily)

while $\pi_k \neq \pi_{k-1}$ do

1. Evaluate $\pi_k$: compute the first return times $x^{\pi_k}$ of the nodes:

$$x^{\pi_k} = P^{\pi_k} x^{\pi_k} + 1$$

2. Improve $\pi_k$: greedily switch all free edges that enhance the first return times such that:

$$P^{\pi_{k+1}} x^{\pi_k} \leq P^{\pi_k} x^{\pi_k}$$

$k \rightarrow k + 1$

end
In practice, PRI converges in a linear number of iterations.
In theory, PRI might need an exponential number of iterations to converge.

There are examples on which *Policy Iteration* needs an exponential number of iterations to converge [Fearnley, 2010].
In theory, PRI might need an exponential number of iterations to converge. There are examples on which Policy Iteration needs an exponential number of iterations to converge [Fearnley, 2010]. These examples do not apply to deterministic MDPs or to discounted MDPs with a fixed discount factor.
In theory, PRI might need an exponential number of iterations to converge.

- There are examples on which **Policy Iteration** needs an exponential number of iterations to converge [Fearnley, 2010].

- These examples do not apply to *deterministic MDPs* or to *discounted MDPs with a fixed discount factor*.

- Do they apply to **PageRank Optimization**?
In theory, PRI might need an exponential number of iterations to converge

- There are examples on which Policy Iteration needs an exponential number of iterations to converge [Fearnley, 2010]

- These examples do not apply to deterministic MDPs or to discounted MDPs with a fixed discount factor.

- Do they apply to PageRank Optimization?

⇒ The answer is no, but almost...
Fearnley’s example.
Transform Fearnley’s example into a PRO problem.

Small transition probabilities?

\[ 1 - \frac{1}{(10n+4)2^n} \Rightarrow \]

An exponentially small transition probability can be replaced by a polynomial sized structure with uniform transition probabilities.
Transform Fearnley’s example into a PRO problem.
Small transition probabilities?

$$1 - \frac{1}{(10n+4)2^n} \Rightarrow$$

$$\frac{1}{(10n+4)2^n}$$

An exponentially small transition probability can be replaced by a polynomial sized structure with uniform transition probabilities.
Transform Fearnley’s example into a PRO problem.

Small transition probabilities?

\[ 1 - \frac{1}{(10n+4)2^n} \]

An exponentially small transition probability can be replaced by a polynomial sized structure with uniform transition probabilities.
Transform Fearnley’s example into a PRO problem.

High costs?

\[ (10n + 4)2^n \]
Transform Fearnley’s example into a PRO problem.

High costs?

\[ x \xrightarrow{(10n + 4)2^n} y \]

\[ G_k \equiv \begin{array}{c}
1 \rightarrow 2 \rightarrow 1 \rightarrow \cdots \rightarrow 1 \rightarrow k \rightarrow 1 \rightarrow 1 \rightarrow 1
\end{array} \]

\[ \Rightarrow \begin{cases}
\text{cost} = 2^k + 2^{k-1} \\
\# \text{nodes} = k + 2
\end{cases} \]
Transform Fearnley’s example into a PRO problem.

High costs?

\[ (10n + 4)2^n \]

\[ G_k \equiv \]

\[ H_k \equiv \]

\[ \Rightarrow \begin{cases} cost = 2^k + 2^{k-1} \\ \# nodes = k + 2 \end{cases} \]

\[ \Rightarrow \begin{cases} cost = 2^k \\ \# nodes = \frac{3}{4}k^2 + k + 1 \end{cases} \]
Transform Fearnley’s example into a PRO problem.

High costs?

\[
x \xrightarrow{(10n + 4)2^n} y \quad \Rightarrow \quad x \xrightarrow{1} H_n \xrightarrow{1} H_n \xrightarrow{1} \ldots \xrightarrow{1} H_n \xrightarrow{1} y
\]

\textit{repeated} \ (10n + 4) \ \textit{times}

\[
G_k \equiv \begin{array}{c}
1 \\
\uparrow \\
1 \\
\downarrow \\
1 \\
1 \\
\uparrow \\
2 \\
\downarrow \\
\ldots \\
\downarrow \\
k \\
\downarrow \\
1 \\
\downarrow \\
1 \\
\downarrow \\
1 \\
\rightarrow \quad \Rightarrow \quad \begin{cases}
\text{cost} = 2^k + 2^{k-1} \\
\# \ nodes = k + 2
\end{cases}
\end{array}
\]

\[
H_k \equiv \begin{array}{c}
G_{k-1} \\
\downarrow \\
G_{k-3} \\
\downarrow \\
\ldots \\
\downarrow \\
G_1 \\
\downarrow \\
\rightarrow \quad \Rightarrow \quad \begin{cases}
\text{cost} = 2^k \\
\# \ nodes = \frac{3}{4}k^2 + k + 1
\end{cases}
\end{array}
\]
Transform Fearnley’s example into a PRO problem.
From actions to free edges?

\[ x \rightarrow y \rightarrow z \]

\[ (1 - \varepsilon) \rightarrow \varepsilon^{18} \]
Transform Fearnley’s example into a PRO problem.
From actions to free edges?

\[
\begin{align*}
\text{x} & \rightarrow \text{y} \\
\downarrow & \\
(1 - \varepsilon) (\varepsilon) & \rightarrow \\
\text{x} & \rightarrow \varepsilon \rightarrow \text{y} \\
\downarrow & \\
\text{z} &
\end{align*}
\]
Transform Fearnley’s example into a PRO problem.
From actions to free edges?

$$\begin{align*}
\text{It works for some } \varepsilon < 1/2. \\
\Rightarrow \\
\text{(1 - } \varepsilon) \\
\varepsilon \\
\text{(1 - } \varepsilon) \\
\Rightarrow \\
\end{align*}$$
Transform Fearnley’s example into a PRO problem.

From actions to free edges?

It works for some \( \varepsilon < \frac{1}{2^{O(n^2)}} \)
Transform Fearnley’s example into a PRO problem.
What about zero and negative costs?
Our main result.

Theorem

If $+1$ and $-1$ costs are allowed, then there exists an infinite family of PageRank Optimization problems on which the number of iterations that PI takes is lower bounded by an exponential function of the size of the problem.
PRI runs in polynomial time in some particular cases.

1. If zapping is included in the problem, then PRI runs in weakly polynomial time. Google’s case!

2. If every free edge leaves either the target node or some other node, then PRI runs in strongly polynomial time. The case of a webmaster with one friend...
Take home messages.

- PageRank Optimization modelizes most problems that consist in optimizing centrality.

- PageRank Optimization is polynomially equivalent to MDPs with only positive costs, provided some regularity assumptions.

- PageRank Iteration is efficient in practice for solving large instances.

- Optimizing PageRank is essentially useful when one seeks to improve the ranking and not the absolute value of its PageRank.
Thanks for your attention!