
Learning to Schedule Webpage Updates Using Genetic Programming

Aecio Santos¹, Nivio Ziviani¹, Jussara Almeida¹,
Cristiano Carvalho¹, Edleno Moura², Altigran Silva²

¹ Universidade Federal de Minas Gerais

² Universidade Federal do Amazonas

SPIRE, Jerusalem, October 7th, 2013

Where we are in Brazil



Manaus

Belo Horizonte

The Problem

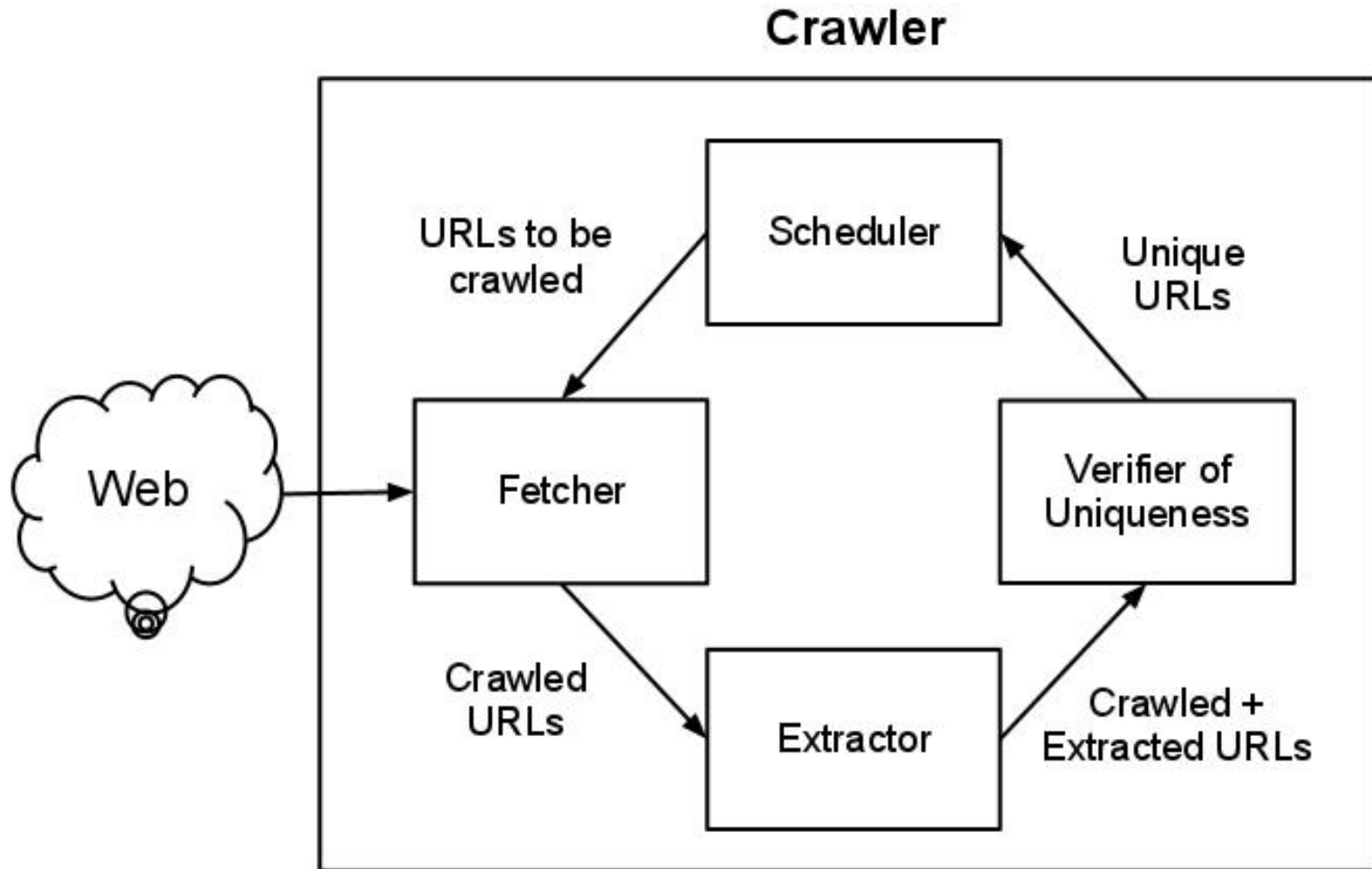
Estimate the likelihood that a webpage has been modified

- This estimation is used to define the order in which those pages should be visited
- Reduce the cost of monitoring crawled webpages for keeping updated versions

Difficulty of the Problem

- Web: over 60 trillion individual webpages
- Scheduler define the order in which URLs should be visited
- Crawlers use a score function to assign a weight to each known URL
- Only the top k pages are taken to be visited
- Full scan of prior crawled webpages to assure freshness is unfeasible

Crawler Architecture: Complete Cycle



The Scheduler

Driven by two main goals:

- *Coverage*: fraction of desired pages downloaded successfully
- *Freshness*: degree to which downloaded pages remain up-to-date
- **This work is focused on freshness**

Related Work

- Estimator for change frequency of pages: visit pages proportionally more often
 - Estimator is used as **baseline**
 - [Cho and Garcia-Molina, ACM TOIT, 2003]
- Group pages into k clusters with similar change behavior and sort them based on the mean change frequency of a sample of each cluster
 - They proposed four strategies to compute weights associated with a change, all used as **baseline**
 - [Tan and Mitra, ACM TOIS, 2010]

GP for Incremental Crawling (GP4C)

Iterative process with two phases:

- Train with an initial set of pages
 - Training set is crawled first
- Validate results with a distinct set of pages
 - A set of validation pages is crawled
 - Fitness function is used in this phase
- Experimental tests apply the resulting function in a third set of pages
- Best individuals are selected as scheduling solutions

Best Individuals in the Validation Step

- Run GP process N times with distinct seeds
- Pick best individual: $GP4C_{Best}$
- Average performance: $GP4C_{Avg}$
- Sum of each individual: $GP4C_{Sum}$
- For $GP4C_{Avg}$ and $GP4C_{Sum}$
 - Considered performances of each individual in both training and validation sets minus SD of such performance when selecting best individuals

Experimental Evaluation

- Crawl simulation to ensure that all policies are compared under same conditions
- Built a dataset collected from the Brazilian Web
 - Repository of around 200 million pages
 - We selected 3,059,698 pages, daily monitored

Monitoring period	Number of webpages	Number of websites	Number of webpages/site		
			Min	Max	Average
57 days	417,048	7,171	1	2,336	58.15

Baselines [Cho and Garcia-Molina, 2003]

$$\text{CG} = -\log\left(\frac{n - X + 0.5}{n + 0.5}\right)$$

- CG: estimates the change frequency of p pages
- n : number of times page p was visited
- X : # of times a page p changed in n visits

Baselines [Tan and Mitra, 2010]

$$\lambda_p = \sum_{i=1}^n w_i \cdot I_i(p)$$

- λ_p : parameter of a Poisson process
 - assuming each page p follows a Poisson process
- n : number of visits
- w_i : weight of change in the i^{th} download
 - NAD, SAD, AAD, GAD
- $I_i(p)$: 1 if page p changed; 0 otherwise

Baselines [Tan and Mitra, 2010]

$$\lambda_p = \sum_{i=1}^n w_i \cdot I_i(p)$$

- w_i : weight of change in the i^{th} download
 - NAD (Nonadaptive): all change events have equal weight
 - SAD (Shortsighted adaptive): considers the current change status of the page
 - AAD (Arithmetically adaptive)
 - GAD (Geometrically adaptive)

Evaluation Metric

- **ChangeRate** at cycle i is the fraction of pages downloaded that were changed

$$C_i = \frac{D_i^c}{D_i}$$

- ChangeRate used as **fitness function**
- [Douglass et al., Rate of change and other metrics: a live study of the world wide web, USENIX 1997]

Experimental Methodology

- 5-fold cross validation
 - 4 folds equally divided into training set and validation set and 5th fold as test set
- Simulate a crawl using dataset to evaluate score functions and compute fitness values

Experimental Methodology

- GP framework:

- $N_p = 300$ individuals
- $N_g = 50$ generations as termination criterion
- Maximum tree depth = 10

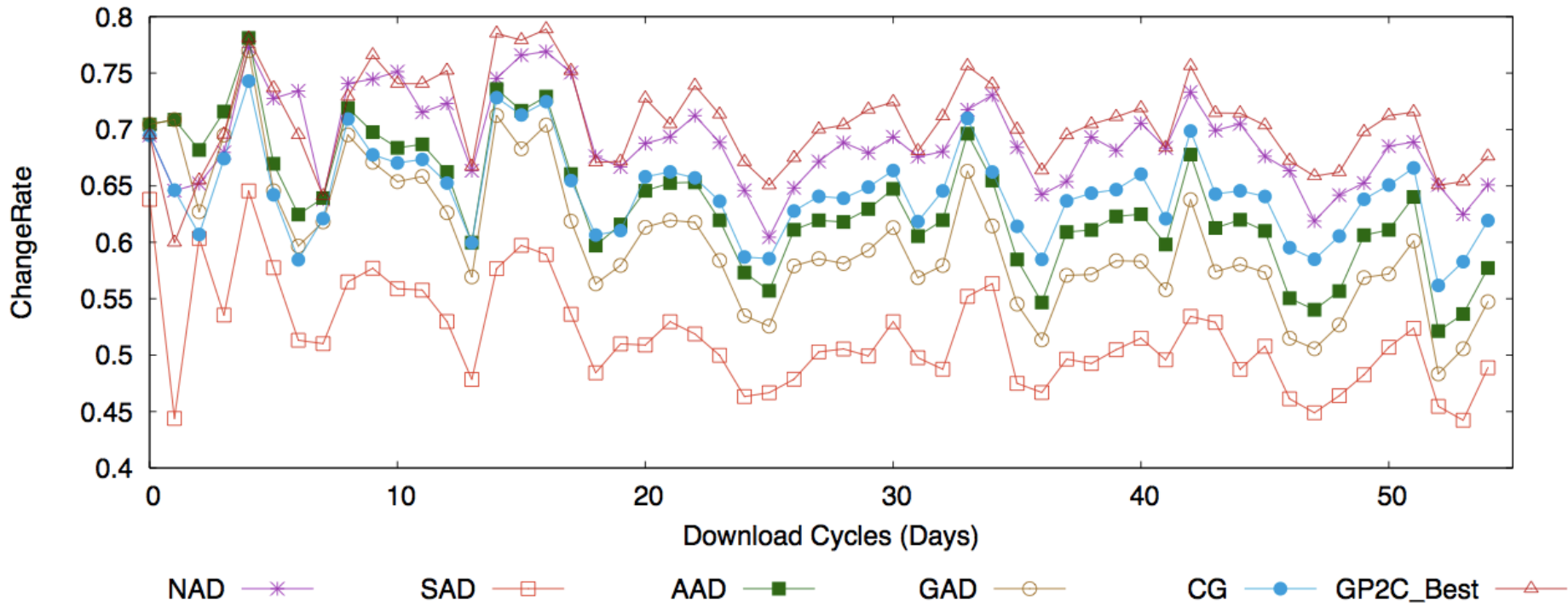
- Terminals:

n : # of times page p was visited

X : # of times page p changed in n visits

t : # of cycles since page p was last visited

Average ChangeRate on Each Cycle



Average ChangeRate for All Days

Rand	Age	NAD	SAD	AAD	GAD	CG	$GP4C_{Best}$	$GP4C_{Sum}$	$GP4C_{Avg}$
0.1857	0.2130	0.6892	0.5166	0.6344	0.6016	0.6439	0.7058	0.7008	0.7034
±	±	±	±	±	±	±	±	±	±
0.0007	0.0009	0.0056	0.0066	0.0095	0.0059	0.0067	0.0096	0.0176	0.0107

Conclusions

- GP framework to automatically generate score functions for schedulers
- Rank webpages according to their likelihood of being modified since last crawled
- Compared three variations of GP4C against seven state-of-the-art baselines
- GP4C_{Best} is statistically superior to all baselines
- Framework quite flexible to (as future work)
 - derive new score functions (e.g., Pagerank of pages)
 - alternative fitness functions to balance coverage and freshness

-
- Aécio Santos, Nivio Ziviani, Jussara Almeida, Cristiano Carvalho, Edleno Moura, Altigran Silva, Learning to Schedule Webpage Updates Using Genetic Programming, Spire 2013, 759-764.
 - Aécio Santos, Cristiano Carvalho, Jussara Almeida, Edleno Moura, Altigran Silva, Nivio Ziviani, A Genetic Programming Framework to Schedule Webpage Updates, Information Retrieval Journal 18(1), 2015, 73-94.

Nivio Ziviani

nivio@dcc.ufmg.br

www.dcc.ufmg.br/~nivio