uOttawa

L'Université canadienne
Canada's university

Software Security Research Group (SSRG)
University of Ottawa
In collaboration with IBM

Dist-RIA Crawler: A Distributed Crawler for Rich Internet Applications

Seyed M. Mirtaheri, Di Zou, **Gregor v. Bochmann,**
Guy-Vincent Jourdan, and Iosif Viorel Onut

8th International Conference on P2P, Parallel, Grid,
Cloud and Internet Computing (3PGCIC)

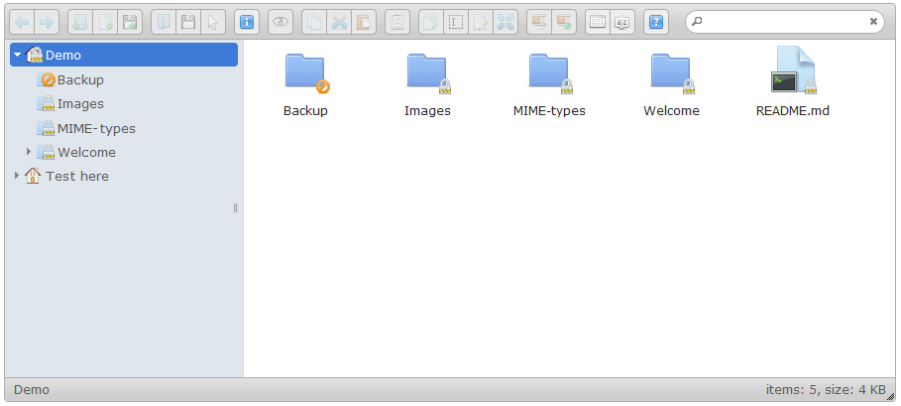
October 28, 2013

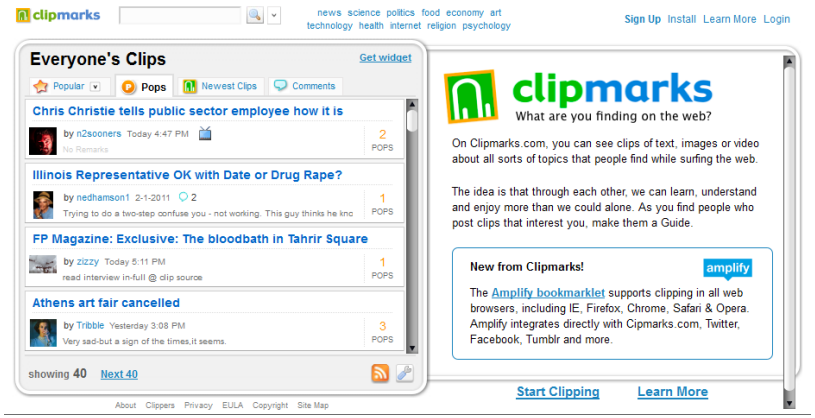
Overview

- Introduction
  - The evolution of the Web Crawling
- Crawling RIAs
  - Challenges and Assumptions
  - Parallel Crawling
- Our Approach – Distributed Crawling
  - Overview
  - Partitioning Algorithm
- Experimental Results
- Future Work

Introduction - Rich Internet Applications

- RIAs shift parts of the computation to the client
- Client side code changes the “page” - the Document Object Model (DOM).
- Events : Occurrences that cause code execution (mouse click, timeout etc.)





Introduction - Crawling

- Crawling: Automatic exploration of the application
- Motivations
  - Content indexing (by search engines)
  - Testing (for security and accessibility)
- Objectives
  - Find all (or 'important') pages
  - Find connections between the pages (page ranking and obtaining a complete model of the application)
- Crawling extracts "a model" of the application
  - **States** are the "distinct" pages
  - **Transitions** are the connections between the states

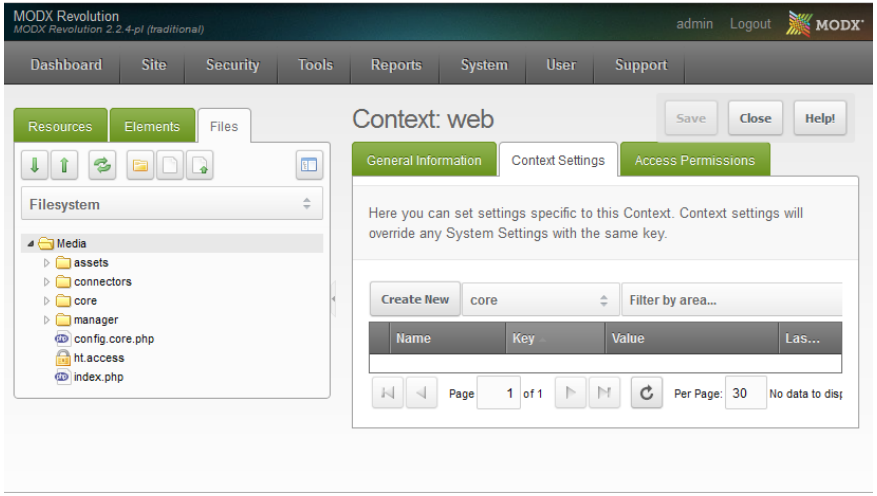
Introduction – chronology of web crawling:

1. Traditional crawling:
   - Every URL is mapped to a single state
   - Crawling is finding all URLs
2. Deep-web crawling:
   - HTML forms are used to access data
   - Crawling involves assigning values to open fields
3. RIA web crawling:
   - Client side events are used to modify the DOM
   - Crawling involves executing all events in each state

Crawling RIAs - Challenges

- State Identification
  - DOM equivalence
  - Detecting independent widgets
- Event Identification
- Deterministic Behaviour (No Server-side States)
- Intermediate States
- Performance

- **Our Focus:**
  Performance



Crawling RIAs - Parallel Crawling

- Parallel crawling in traditional Web Apps
  - WebCrawler & MOMspider
    - First parallel crawlers
  - Google
    - PageRank, Compression, etc
  - Mercator, Polybot, pSearch, IRLbotpages
    - Focus on URL-Seen
- Very little work on Parallel Crawling of RIAs:
  - 2011, Mesbah et al.
    - Multi-threading, shared memory

Our Approach – Overview

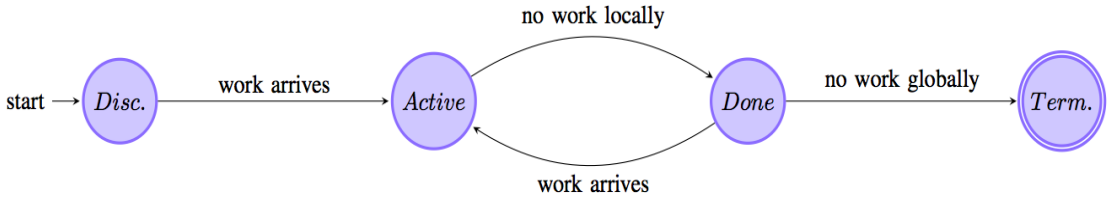
- Executing events is time consuming
- We (statically) distribute the responsibility of executing client-side events among nodes
- Together, all nodes execute all events on all states and discover the entire model of the application.

Our Approach - Partitioning algorithm

- Each node goes to every state. However, each node executes only some of the events.
- The partitioning algorithm decides which are the events that are executed by each node.
  - All events should be executed
  - No work duplications
- Examples:
  - Range based partitioning
  - Stride based partitioning
  - Hash based partitioning

Our Approach – Crawling Algorithm

```
GETCREDENTIALSFROMCOORDINATOR()
NODESTATUS ← ACTIVE
while (NODESTATUS is not TERMINATED) do
    if WORKINGSTATES is Empty then
        GETNEWSTATESFROMCOORDINATOR()
        if WORKINGSTATES is Empty then
            NODESTATUS ← DONE
            SENDNODESTATUSTOCOORDINATOR()
        else
            NODESTATUS ← ACTIVE
        end if
    else
        stateToVisit ← PICKSTATE(WORKINGSTATES)
        eventToExecute ← PICKUNEXECUTEDEVENT(stateToVisit)
        EXECUTEEVENT(stateToVisit, eventToExecute)
        if CURRENTSTATE is not in DISCOVEREDSTATES then
            push CURRENTSTATE to DISCOVEREDSTATES
            push ASSIGN(CURRENTSTATE) to CURRENTSTATE.UNEXECUTEDEVENTS
            push CURRENTSTATE to WORKINGSTATES
            SENDNEWSTATETOCOORDINATOR( CURRENTSTATE )
        end if
        stateToVisit.REMOVEUNEXECUTEDEVENT(eventToExecute)
        if stateToVisit.UNEXECUTEDEVENTS is empty then
            WORKINGSTATES.REMOVESTATE(stateToVisit)
        end if
    end if
end while
```



Experimental Results

- Dist-RIA Crawler: Implemented on a prototype of IBM® Security AppScan®.
- System parameters :
  - Breadth-First crawling strategy
  - Static partitioning algorithm (no dynamic load balancing)
  - Star communication topology
  - 15 nodes + 1 coordinator

Target application

- RIA File Tree browser

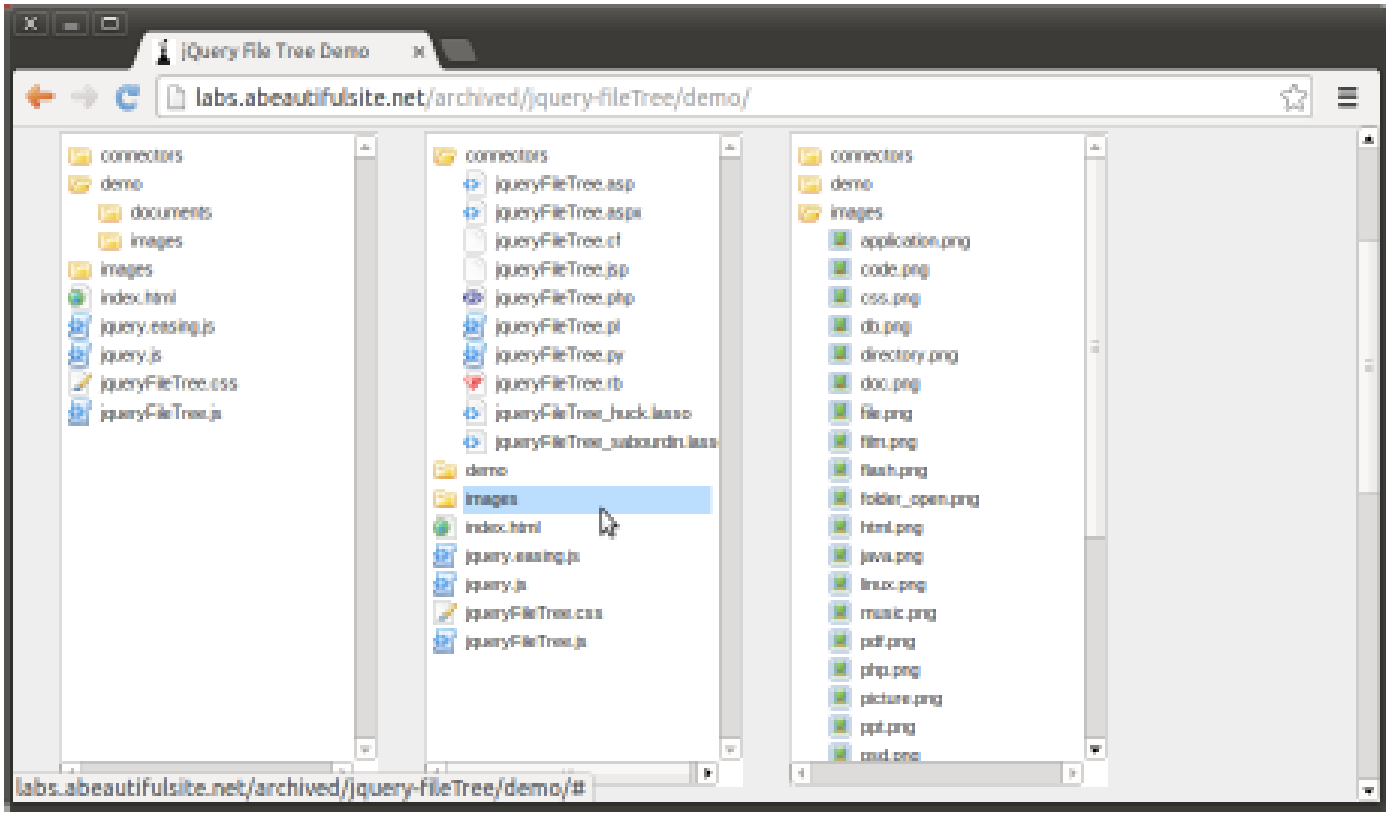


Fig. 2: File tree browser RIA screen-shot

Dist-RIA Crawler: Time to crawl two RIAs

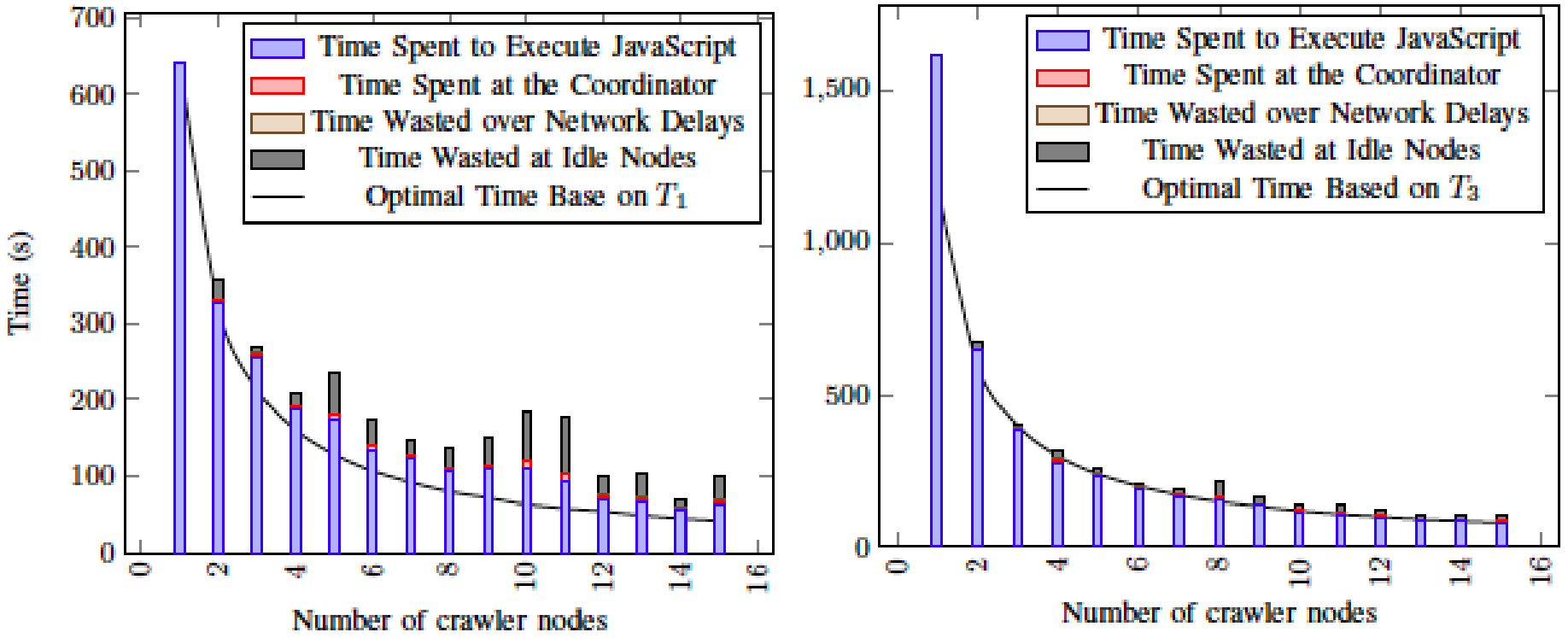
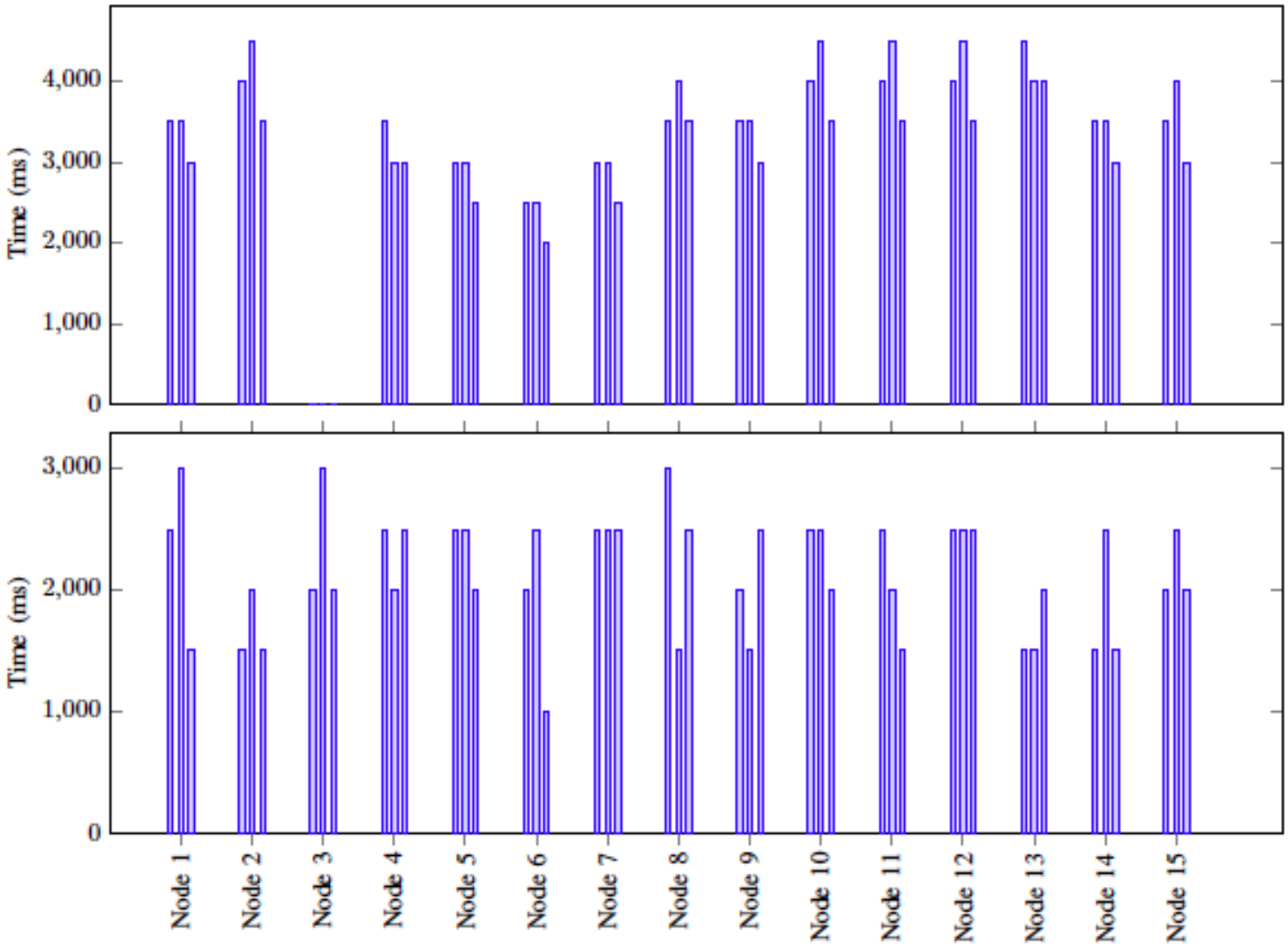


Fig. 3: Time to crawl a RIA with multiple nodes: Apache HTTPD source code file browser (left), and Apache Cassandra source code file browser (right).

Dist-RIA Crawler: Idle times



Conclusions

- Performance improvement (essentially) proportional to number of nodes
- Need for dynamic load balancing to avoid idling of nodes (towards the end of a crawl)
- A single coordinator can handle up to approximately 50 nodes
- Ongoing work:
  - Dynamic load balancing
  - Other crawling strategies
  - Other event partitioning algorithms
  - Avoiding single coordinator : Using a P2P setting

References

[1] D Turgay Altilar and Yakup Paker. Optimal scheduling algorithms for communica- tion constrained parallel processing, 2002.

[2] Domenico Amalfitano, Anna Rita Fasolino, and Porfirio Tramontana. Reverse engi- neering finite state machines from rich internet applications. In Proceedings of the 2008 15th Working Conference on Reverse Engineering, WCRE '08, pages 69–73, Washington, DC, USA, 2008. IEEE Computer Society.

[3] Domenico Amalfitano, Anna Rita Fasolino, and Porfirio Tramontana. Experiment- ing a reverse engineering technique for modelling the behaviour of rich internet applications. In Software Maintenance, 2009. ICSM 2009. IEEE International Con- ference on, pages 571 –574, sept. 2009.

[4] Domenico Amalfitano, Anna Rita Fasolino, and Porfirio Tramontana. Rich internet application testing using execution trace data. In Proceedings of the 2010 Third International Conference on Software Testing, Verification, and Validation Work- shops, ICSTW '10, pages 274–283, Washington, DC, USA, 2010. IEEE Computer Society.

[5] Domenico Amalfitano, Anna Rita Fasolino, and Porfirio Tramontana. Techniques and tools for rich internet applications testing. In Web Systems Evolution (WSE), 2010 12th IEEE International Symposium on, pages 63 –72, sept. 2010.

[6] Amnon Barak. The mosix multicomputer operating system for high performance cluster computing. Journal of Future Generation Computer Systems, 13:4–5, 1998.

[7] J. Barbosa, J. Tavares, and A. J. Padilha. Linear algebra algorithms in hetero- geneous cluster of personal computers. In Proceedings of the 9th Heterogeneous Computing Workshop, HCW '00, pages 147–, Washington, DC, USA, 2000. IEEE Computer Society.

[8] Luciano Barbosa and Juliana Freire. Siphoning hidden-web data through keyword- based interfaces. In In SBBD, pages 309–321, 2004.

[9] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In Proceedings of the nineteenth ACM symposium on Operating systems principles, SOSP '03, pages 164–177, New York, NY, USA, 2003. ACM.

References

[10] Jason Bau, Elie Bursztein, Divij Gupta, and John Mitchell. State of the art: Au- tomated black-box web application vulnerability testing. In Security and Privacy (SP), 2010 IEEE Symposium on, pages 332–345. IEEE, 2010.

[11] Kamara Benjamin. A strategy for efficient crawling of rich internet applications. Master's thesis, EECS - University of Ottawa, 2010. http://ssrg.eecs.uottawa. ca/docs/Benjamin-Thesis.pdf.

[12] Kamara Benjamin, Gregor Von Bochmann, Mustafa Emre Dincturk, Guy-Vincent Jourdan, and Iosif Viorel Onut. A strategy for efficient crawling of rich internet applications. In Proceedings of the 11th international conference on Web engineering, ICWE'11, pages 74–89, Berlin, Heidelberg, 2011. Springer-Verlag.

[13] M. J. Berger and S. H. Bokhari. A partitioning strategy for nonuniform problems on multiprocessors. IEEE Trans. Comput., 36(5):570–580, May 1987.

[14] Michael K. Bergman. The deep web: Surfacing hidden value. September 2001.

[15] Paolo Boldi, Bruno Codenotti, Massimo Santini, and Sebastiano Vigna. Ubicrawler: A scalable fully distributed web crawler. Proc Australian World Wide Web Confer- ence, 34(8):711–726, 2002.

[16] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In Proceedings of the seventh international conference on World Wide Web 7, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[17] Thang Nguyen Bui and Curt Jones. A heuristic for reducing fill-in in sparse matrix factorization. In PPSC, pages 445–452, 1993.

[18] Mike Burner. Crawling towards eternity: Building an archive of the world wide web. Web Techniques Magazine, 2(5), May 1997.

[19] Brent Callaghan, Theresa Lingutla-Raj, Alex Chiu, Peter Staubach, and Omer Asad. Nfs over rdma. In Proceedings of the ACM SIGCOMM workshop on Network- I/O convergence: experience, lessons, implications, NICELI '03, pages 196–208, New York, NY, USA, 2003. ACM.

[20] Duen Horng Chau, Shashank Pandit, Samuel Wang, and Christos Faloutsos. Parallel crawling for online social networks. In Proceedings of the 16th international con- ference on World Wide Web, WWW '07, pages 1283–1284, New York, NY, USA, 2007. ACM.

References

[20] Duen Horng Chau, Shashank Pandit, Samuel Wang, and Christos Faloutsos. Parallel crawling for online social networks. In Proceedings of the 16th international con- ference on World Wide Web, WWW '07, pages 1283–1284, New York, NY, USA, 2007. ACM.

[21] Junghoo Cho and Hector Garcia-Molina. Parallel crawlers. Technical Report 2002-9, Stanford InfoLab, February 2002.

[22] Suryakant Choudhary. M-crawler: Crawling rich internet applications using menu meta-model. Master's thesis, EECS - University of Ottawa, 2012. http://ssrg. site.uottawa.ca/docs/Surya-Thesis.pdf.

[23] Suryakant Choudhary, Mustafa Emre Dincturk, Gregor von Bochmann, Guy- Vincent Jourdan, Iosif-Viorel Onut, and Paul Ionescu. Solving some modeling chal- lenges when testing rich internet applications for security. In ICST, pages 850–857, 2012.

[24] Suryakant Choudhary, Mustafa Emre Dincturk, Seyed M. Mirtaheri Gregor von Bochmann, Guy-Vincent Jourdan, and Iosif-Viorel Onut. Crawling rich internet applications: The state of the art. In Proceedings of the 2012 Conference of the Center for Advanced Studies on Collaborative Research, CASCON '12, Riverton, NJ, USA, 2012. IBM Corp.

[25] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. Introduction to Algo- rithms. The MIT Press, 3rd edition, 2009.

[26] G. Cybenko. Dynamic load balancing for distributed memory multiprocessors. Jour- nal of Parallel and Distributed Computing, 7(2):279–301, October 1989.

[27] L. Dagum and R. Menon. OpenMP: an industry standard API for shared-memory programming. IEEE Computational Science and Engineering, 5(1):46–55, 1998.

[28] Karen D. Devine, Erik G. Boman, Robert T. Heaphy, Bruce A. Hendrickson, James D. Teresco, Jamal Faik, Joseph E. Flaherty, and Luis G. Gervasio. New challenges in dynamic load balancing. APPL. NUMER. MATH, 52:2005, 2004.

[29] Mustafa Emre Dincturk, Suryakant Choudhary, Gregor Von Bochmann, , Guy- Vincent Jourdan, and Iosif Viorel Onut. A statistical approach for efficient crawling of rich internet applications. In Proceedings of the 12th international conference on Web engineering, ICWE'12, pages 74–89, Berlin, Heidelberg, 2012. Springer-Verlag.

Any Questions ??