ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Multi-level, Multi-core Distributed Trace Synchronization

Masoume Jabbarifar
Masoume.jabbarifar@polymtl.ca

Supervisor: Michel Dagenais

DORSAL

9 December 2011

Outline

- Introduction
- Online Synchronization Approaches
- Results
- Conclusion
- References

Streaming Data Challenges

- Synchronizing a live trace stream on the fly.
  - It is not practical to scan the data stream more than once
  - Buffering the data stream for a long period is problematic

Goal 1: *Online time synchronization of distributed traces has to be efficient in both time and memory*

Goal 2: *Prevent reading the whole data from the start point of tracing to the end of the current time*

Goal 3: *Online time synchronization has to be scalable and should not lose the accuracy over time*

Time Window based approaches

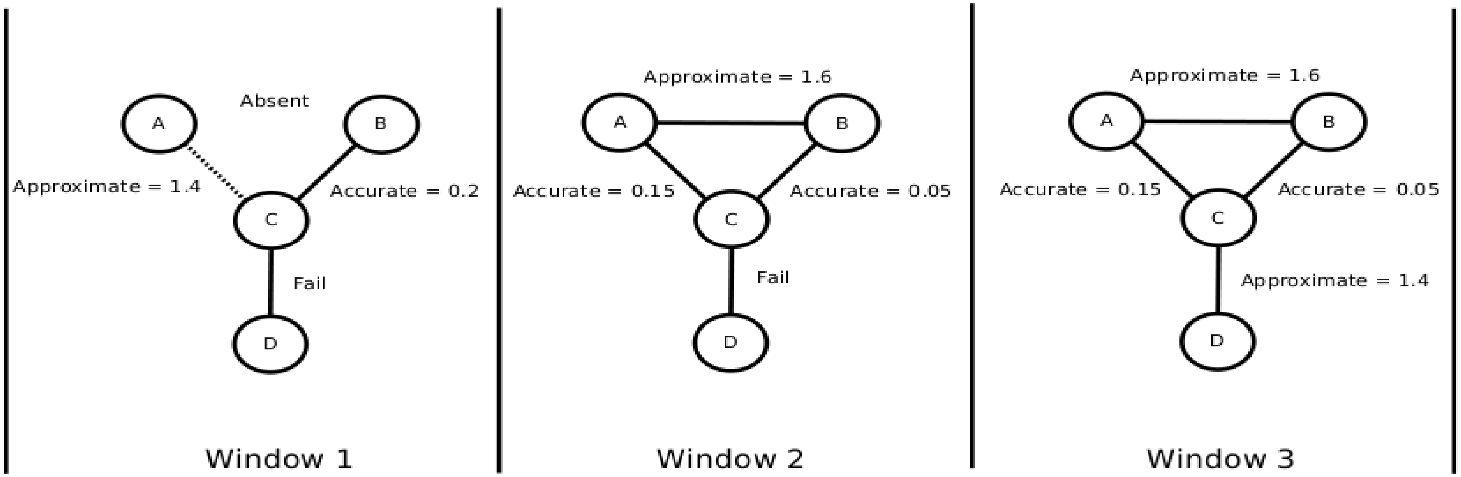
- Independent
- Replace
- Merge (10%, 50%, and 90%)
- Correlated

Independent Approach

- Analyze one time window at a time, independently
- Advantages:
  - No buffering or dependency on data from previous time windows.
  - Simpler and more efficient compared with the three other approaches
- Disadvantages:
  - It is not able to achieve a satisfactory accuracy, not only in each window, but also after a settling period.

Replacement Approach

- Using the useful results from convex-hulls of previous windows
- This approach insures accuracy improvement over time but the rate of improvement is slow



```
For each link
    d = Accuracy(i–1) – Accuracy(i)
    if d > Threshold
        replace
    else
        drop current result
```

Merging Approach

- Merging the synchronization results of current window and previous window

$$Accuracy\ (i) = k * Accuracy\ (i-1) + (1-k) * a(i)$$

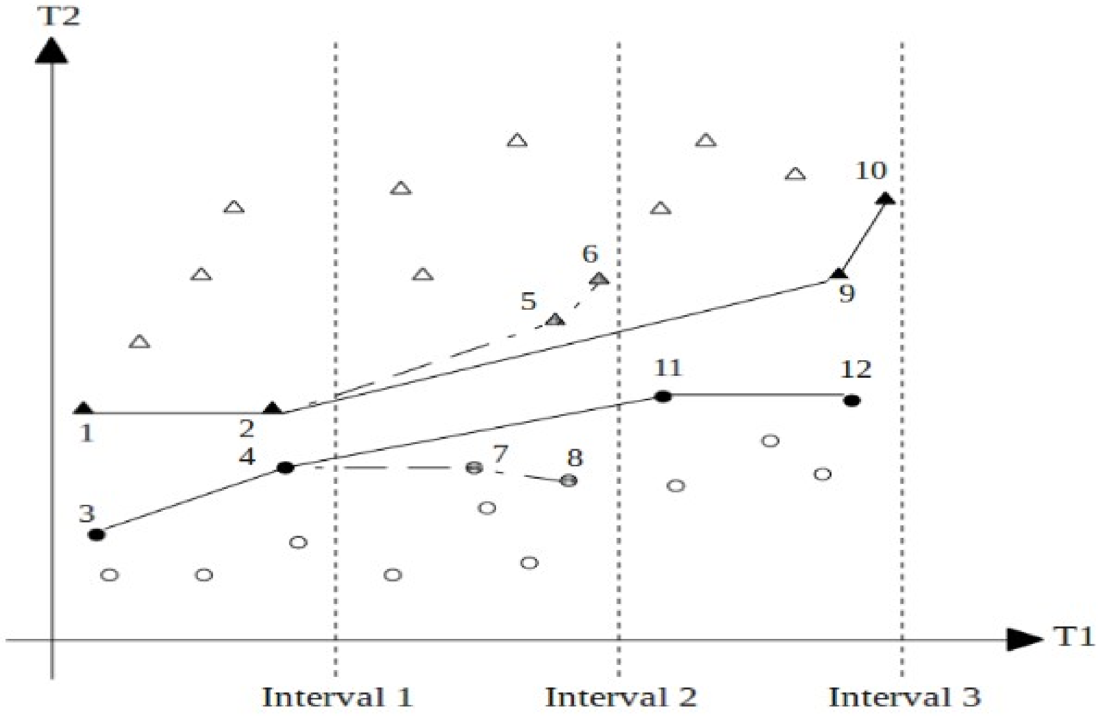
- *a(i): the output of the Convex-hull algorithm for window i*

- *k: the weighting factor*

- Different approaches can be defined based on *k* value:
  - Merge10 (K = 10)
  - Merge50 (K = 50)
  - Merge90 (K = 90)

Correlated Approach

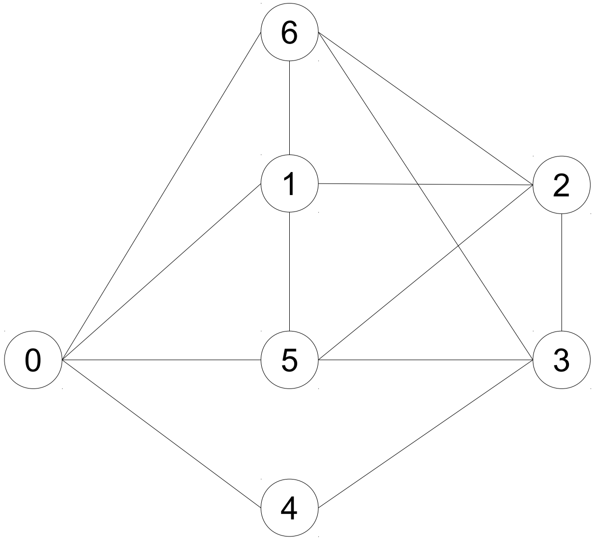
- Select the accurate packets in each window and transfer them to the next window



Correlated Sliding Window

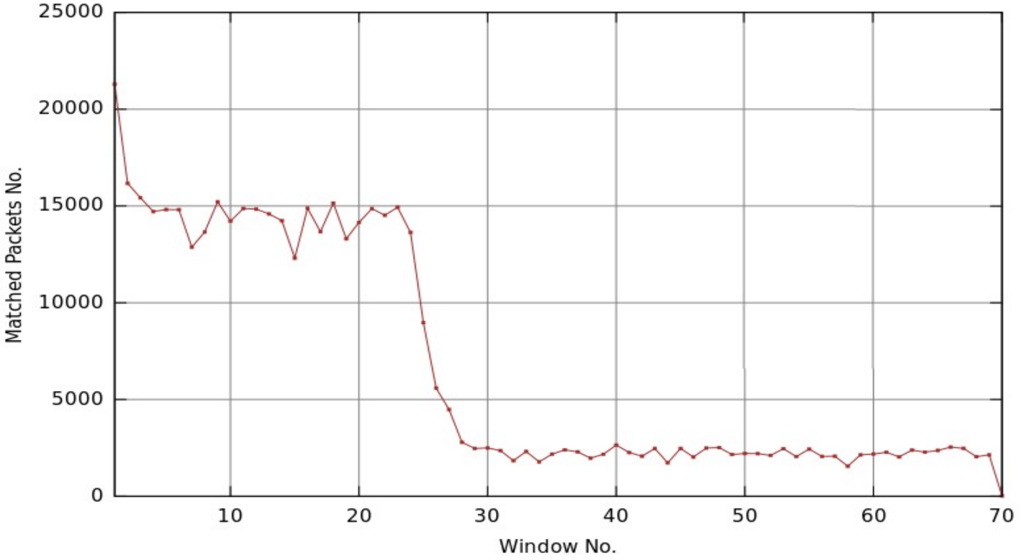
Cluster Setup

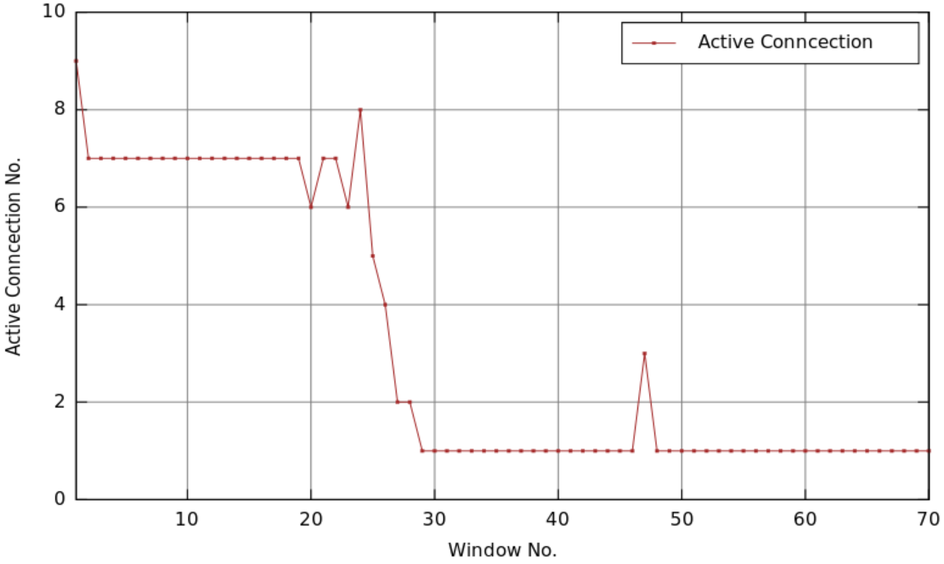
- Synchronisation of 7 nodes
- Each node is a Pentium III (4 CPUs) with 4 GB of RAM
- Window size is 3 seconds
- The nodes relationship graph:



Test Results (1)

- Total input and output events matched together to form a packet in each window

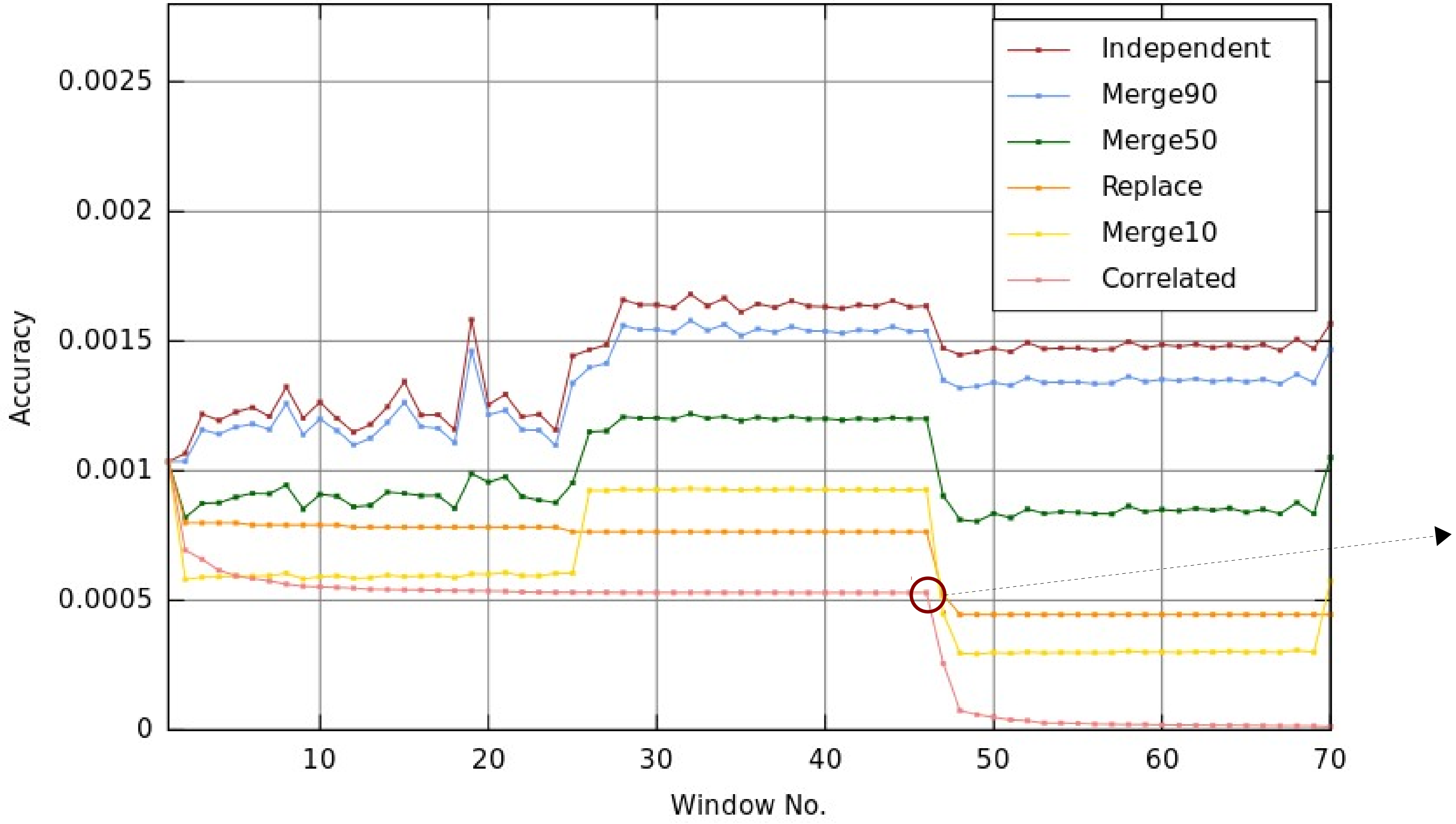




- The number of active connections in each window

Test Results (2)

- The best approach is correlated sliding window



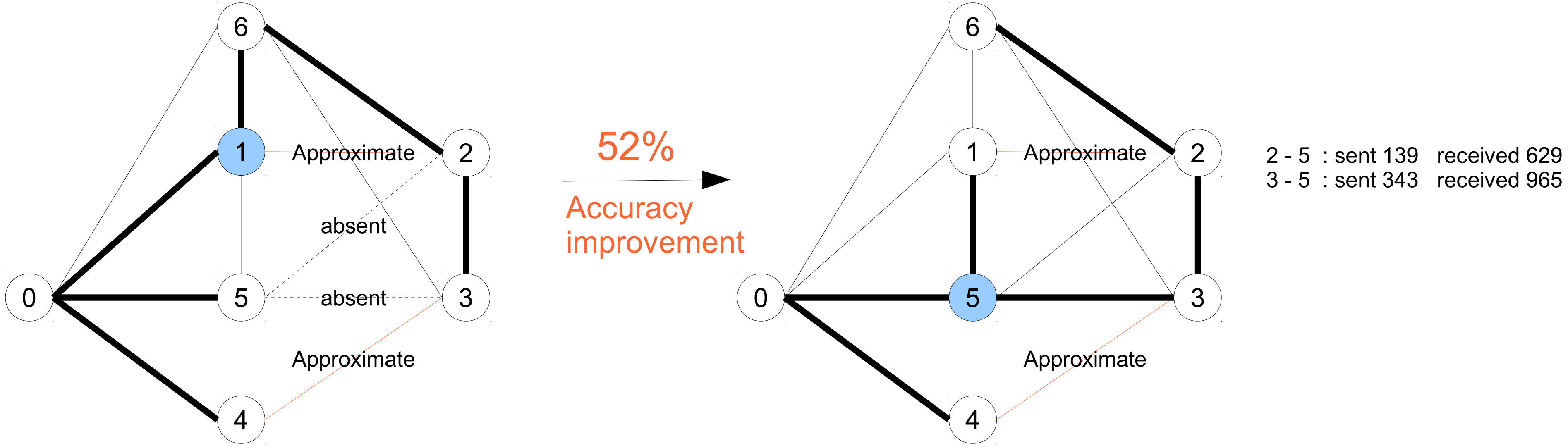
Why does accuracy change very rapidly?

Test Results (3)

- From window 1 to window 46, Node 1 is reference node
- From window 47 to window 70, Node 5 is reference node

System state before 47th Synchronization

System state in 47th Synchronization



52%

Accuracy improvement

2 - 5 : sent 139 received 629
3 - 5 : sent 343 received 965

(a)

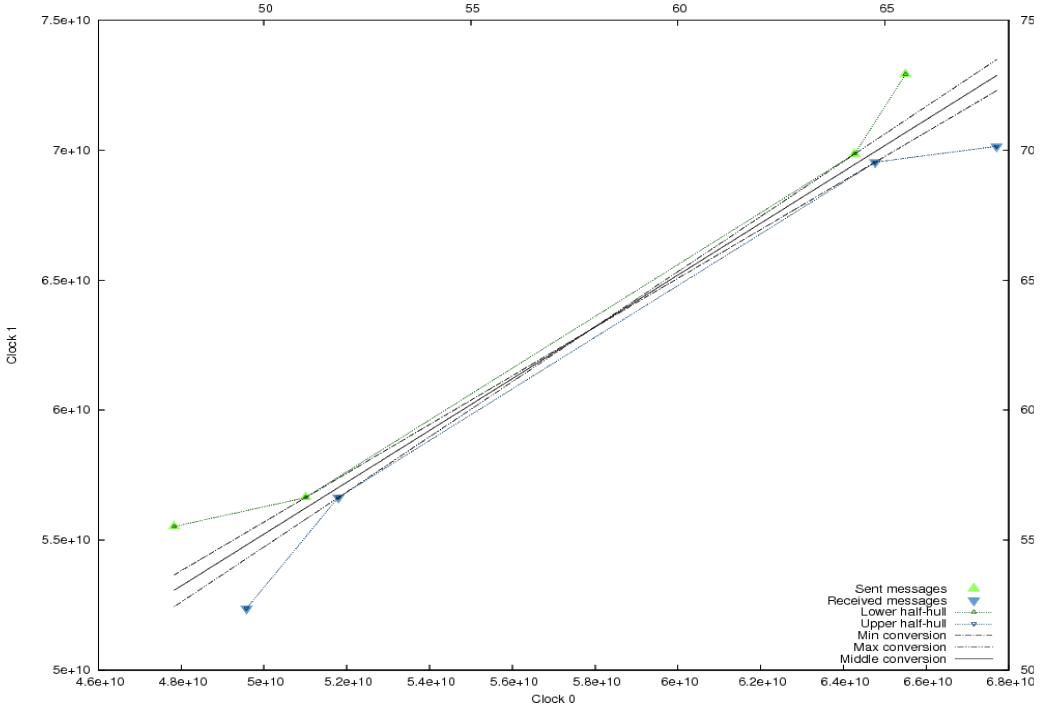
(b)

Question?

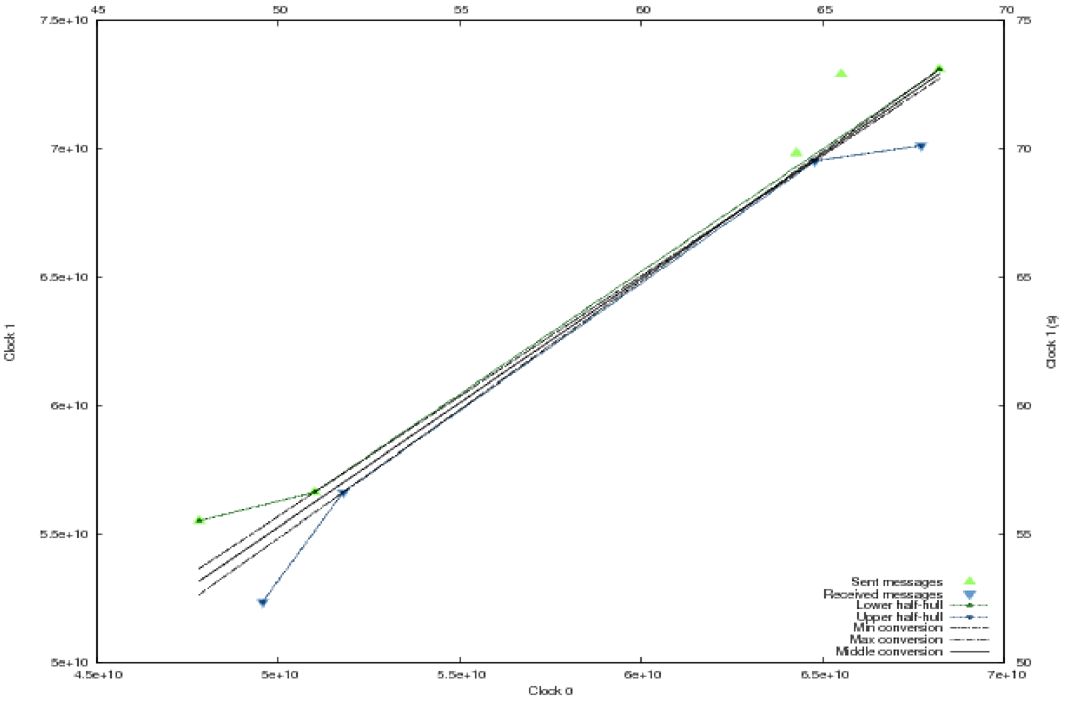
Is there any way to improve the correlated sliding window technique?

Incremental approach (1)

- The lowest distance to the middle line is the best accurate packet
- Accurate packets improve accuracy



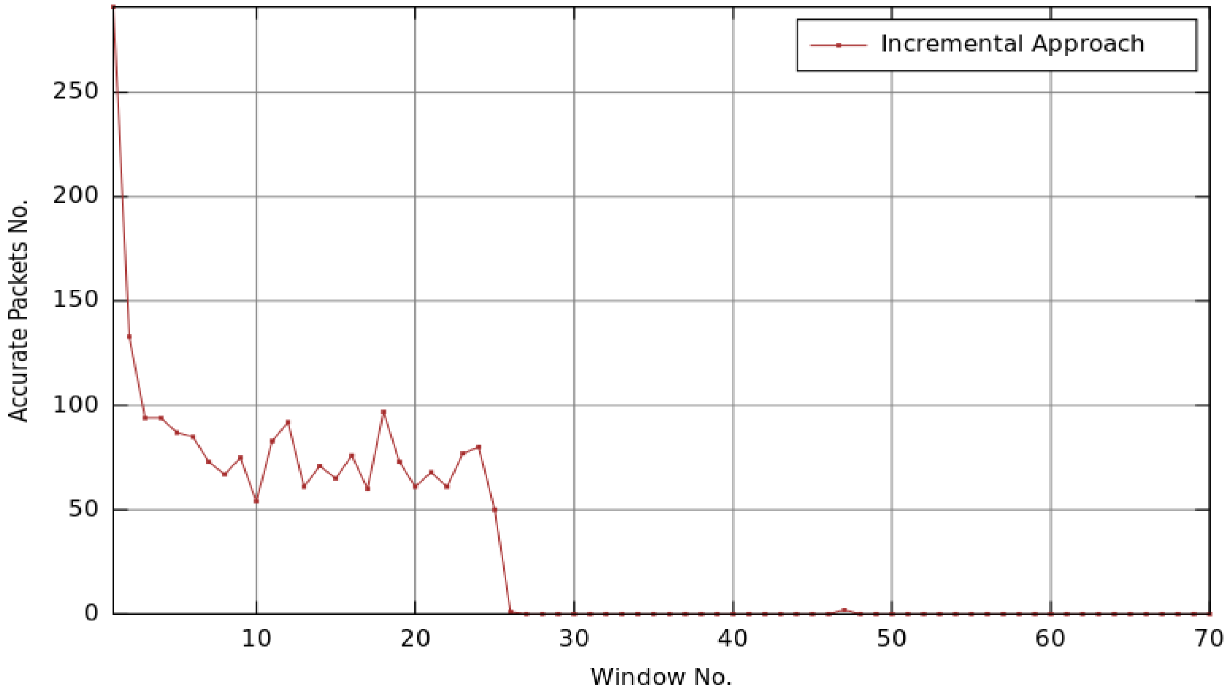
(a) Accuracy : 0.121842



(b) Accuracy : 0.067378

Incremental approach (2)

- There are many accurate packets between window 1 and 26 because there are many active connections
- If we recompute the synchronization each time an accurate packet is received, it increases the analysis time

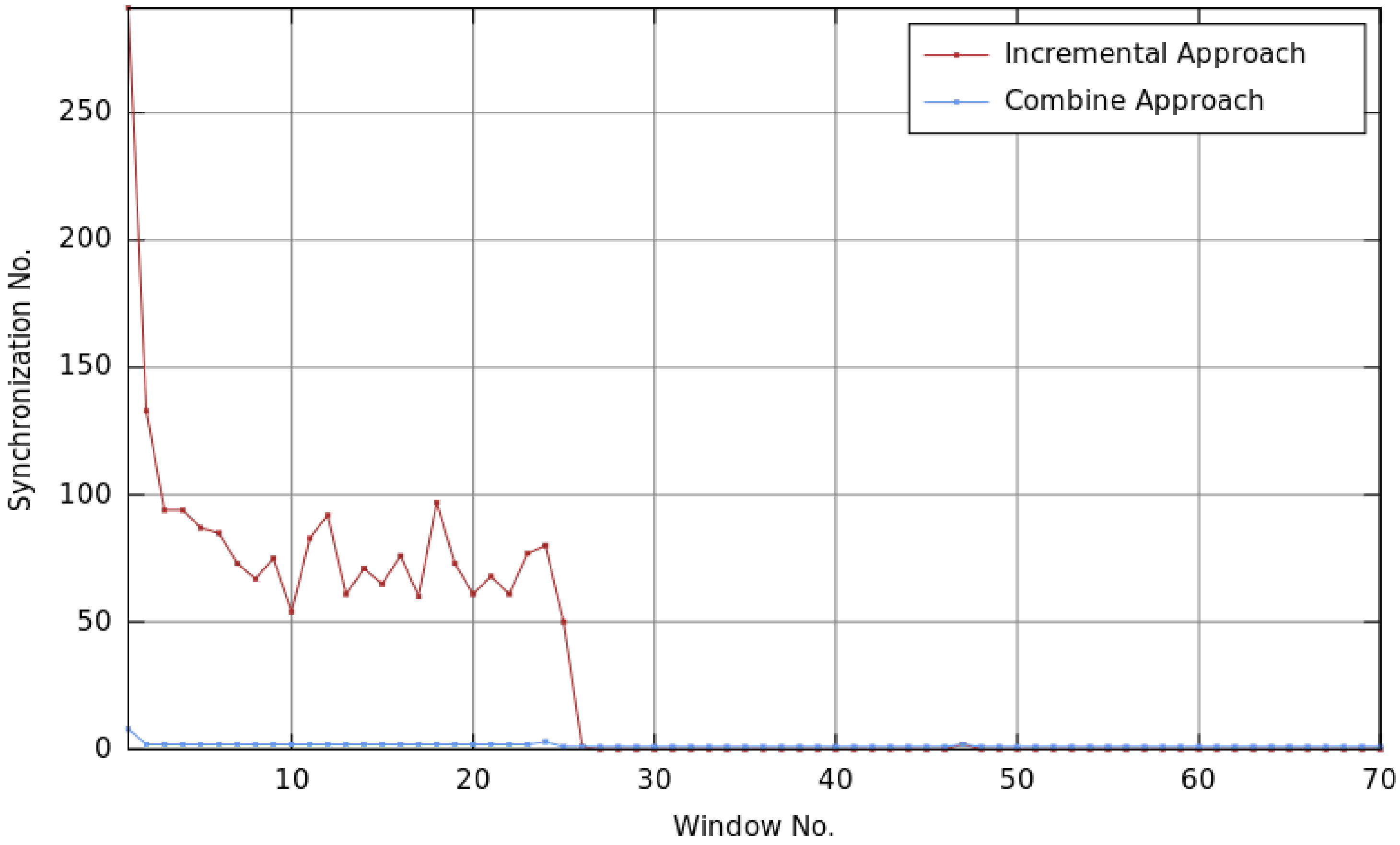


Incremental approach (3)

- Criteria to manage the accurate packets
  - Add window technique
  - Each link has <u>a chance</u> to activate time synchronization in each window
  - Synchronizing at the end of window if we were not triggered by an accurate packet
    - Sometimes a packet does not have minimum distance to the conversion functions but removes some packets on upper or lower hull (***interesting packet***)
    - Interesting packets improve accuracy a little
    - There is a trade off between the <u>*cost of synchronization*</u> and the accuracy increase we get with an interesting packet

Combined approach

- Combined approach has the same accuracy as correlated approach



Conclusion

- Live trace synchronization is ready for deployment.
- The combined approach offers an excellent compromise between performance and accuracy.

References (1)

[1] B. Poirier, R. Roy, and M. Dagenais, “Accurate Offline Synchronization of Distributed Traces Using Kernel-level Events," Operating Systems Review, vol. 44, 2010, pp. 75-87.

[2] J. H. Deschenes, M. Desnoyers and M. Dagenais. “Tracing Time Operating System State Determination,” The Open Software Engineering Journal, vol. 2, 2008, pp. 40-44.

[3] A. Duda, G. Harrus, Y. Haddad, and G. Bernard, “Estimation global time in distributed system," In proceeding 7th Int. Conf. on Distributed Computing Systems, Berlin, volume 18, 1987.

[4] S.B. Moon, P. Skelly, D. Towsley, “Estimation and Removal of Clock Skew from Network Delay Measurements," in: INFOCOM, 1999.

[5] H. Khlifi and J. C. Gregorie, "Low-complexity offline and online clock skew estimation and removal," The International Journal of Computer and Telecommunications Networking, vol. 50, no. 11, pp. 1872-1884, 2006.

[6] A. D. Ksehmkalyani and M. Singhal, “Logical time,” in Distributed Computing: Principles, Algorithms, and Systems, 1st ed., USA: Cambridge University Press, 2008, pp. 50-84.

[7] M. Bligh, M. Desnoyers, and R. Schultz, “Linux kernel debugging on google-sized clusters," In Proceedings of the Linux Symposium, 2007.

[8] M. Desnoyers, “Low-Impact Operationg System Tracing," PhD thesis, Ecole Polytechnique de Montreal, 2009.

[9] R. Sirdey and F. Maurice, “A linear programming approach to highly precise clock synchronization over a packet network,” 4OR: A Quarterly Journal of Operations Research, vol. 6, no. 4, 2008, pp. 393-401.

[10] B. Scheuermann, W. Kiess, M.Roos, F. Jarre, and M. Mauve, “On the time synchronization of distributed log files in networks with local broadcast media," IEEE/ACM Transactions on Networking, vol. 17 no.2, 2009, pp. 431-444.

[11] J. Jezequel, and C. Jard, “Building a global clock for observing computations in distributed memory parallel computers," Concurrency: Practice and Experience, vol. 8 no.1, 1996.

References (2)

[12] H. Marouani, and M.R. Dagenais, “Internal Clock Drift Estimation in Computer Clusters," Journal of Computer Systems, Networks, and Communications, vol.2008 no. 1, 2008, pp. 1-7.

[13] L.M. He, “Time Synchronization Based on Spanning Tree for Wireless Sensor Networks," 4th International Conference on Wireless Communications, Networking and mobile Computing, Dalian, 2008, pp. 1-4.

[14] Mammoth project available at “https://rqchp.ca/?mod=cms&pageId=566&lang=EN," Sep. 2010.

[15] L. Chai, Q. Gao and D. K. Panda, “Understanding the Impact of Multi-Core Architecture in Cluster Computing" A Case Study with Intel Dual-Core System,” Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid, Rio De Janeiro, Brazil, 2007, pp. 471-478.

[16] J. M. Jezequel and C. Jard, “Building a global clock for observing computations in distributed memory parallel computers,” Concurrency: Practice and Experience, vol 2, no. 1, 1996, pp. 71-89

[17] E. Betti, M. Cesati, R Gioiosa and F. Piermaria, “A global operating system for HPC clusters,” IEEE International Conference on Cluster Computing and Workshops, 2009.

[18] C. N. Keltcher, K. J. McGrath, A. Ahmed, and P. Conway, “The amd opteron processor for multiprocessor servers,” IEEE Micro, vol. 23, no. 2, 2003, pp. 66–76.

[19] M. Papakipos, “High-Productivity Software Development for Multi-Core Processors,” 2007. [Online]. Available: http://download.microsoft.com/download/d/f/6/df6accd5-4bf2-4984-8285-f4f23b7b1f37/WinHEC2007_PeakStream.doc [Accessed: 14 April 2010].

[20] NIST Time and frequency from A to Z., February 2011. http://tf.nist.gov/general/glossary.htm.

[21] E. Clement and M. Dagenais, “Trace synchronization in distributed networks,” Journal of computer system, Network, and Communication, 2009.

[22] P. Ashton, “Algorithms for off-line clock synchronization,” Technical report, University of Canterbury, Department of Computer Science, Dec. 1995.

References (3)

[23] J. Doleschal, A. Kn¨pfer, M. S. M¨ller, and W. E. Nagel, “Internal timer synchronization for parallel event tracing,” In Proceedings of the 15th European PVM/MPI Users’ Group Meeting on Recent Advances in Parallel Virtual Machine and Message Passing Interface, pages 202–209, Berlin, Heidelberg, 2008. Springer-Verlag.

[24] K. Berket, R. Koch, L.E. Moser, P.M. Melliar-Smith, “Timestamp acknowledgements for determining message stability,” in: Proceedings of the Second International Conference on Parallel and Distributed Computing and Networks, Brisbane, Australia, December 1998.

[25] B. Scheuermann, W. Kiess, M. Roos, F. Jarre, and M. Mauve, “On the time synchronization of distributed log files in networks with local broadcast media,” Networking, IEEE/ACM Transactions on, 17(2): 431–444, April 2009.

[26] D. Dolev, N. Lynch, S. Pinter, E. Strark, and W. Weihl, “Reaching approximate agreement in the presence of faults,” JACM 33, 3 (July), 499–516, 1986.

[27] J. Joseph, and C. Fellenstein, “Grid Computing,” Prentice Hall, 2003.

[28] K. Iwanicki, “Gossip-based dissemination of time,” Master’s thesis, Warsaw University and Vrije Universiteit Amsterdam (2005).

[29] K. Iwanicki, M. van Steen, S. Voulgaris, “Gossip-based clock synchronization for large decentralized systems,” in: Proc. Workshop on Self-Managed Networks, Systems and Services, vol. 3996 of LNCS, Springer, 2006, pp. 28–42.

[30] M. Jelasity, R. Guerraoui, A. -M. Kermarrec, M. van Steen, “The peer sampling service: Experimental evaluation of unstructured gossip-based implementations,” in: Proc. ACM/IFIP/USENIX Middleware Conf, vol. 3231 of LNCS, Springer, 2004, pp.79–98.

[31] A. -M. Kermarrec, M. van Steen, “Gossiping in distributed systems,” SIGOPS Oper. Syst. Rev. 41 (5) (2007) 2–7.

[32] H. Marouani and M. Dagenais, “Comparing high resolution timestamps in computer clusters," IEEE, 2005.

[33] D. Salyers, A. Striegel, C. Poellabauer, “A Light Weight Method for Maintaining Clock Synchronization for Networked Systems," IEEE, 2008.

References (4)

[34] G. Coulouris, J. Dollimore, T. Kindberg, “ Distributed Systems concepts and design” 4th edition, Addison Wesley, 2005.

[35] A. S. Tanenbaum, M. V. Steen, “Distributed Systems principles and paradigms," 2nd edition, Prentice Hall, 2006.

[36] H.Marouani and M. Dagenais, “Internal Clock Drift Estimation in Computer Cluster," IEEE, 2008.

[37] J. Blunck, M. Desnoyers, and P. -M. Fournier, “Userspace application tracing with markers and tracepoints,” in Proceedings of the 2009 Linux Kongress, Oct. 2009.

[38] M. Desnoyers and M.R. Dagenais, “Deploying LTTng on Exotic Embedded Architectures,” in Embedded Linux Conference 2009, 2009.

[39] Available in http://en.wikipedia.org/wiki/Time_Stamp_Counter in April 2011

[40] M. A. Dietz. “Gathering And Using Time Measurements In Distributed Systems” PhD thesis, Duke University, 1996.

[41] J. Desfossez and M. Dagenais , “Virtual machines traces synchronization” presentation in the Dorsal laboratory, 2010.

[42] M. Jabbarifar, A. S. Sendi, H. Pedram, M. Dehghan and M. Dagenais, “L-SYNC: Larger Degree Clustering Based Time-Synchronisation for Wireless Sensor Network,” Eighth ACIS International Conference on Software Engineering Research, Management and Applications, Montreal, 2010.

[43] M. Jabbarifar, A. S. Sendi, Alireza Sadighian, Naser Ezzati Jivan, and M. Dagenais, “A Reliable and Efficient Time Synchronization Protocol for Heterogeneous Wireless Sensor Network,” Journal of Wireless Sensor Network, vol.2, 2010, pp. 910-918.

[44] M. Jabbarifar, M. Dagenais and R.Roy, “Optimum off-line trace synchronization of computer clusters,” has been sent to HPCS (High Performance Computing Symposium), 2011

References (5)

[45] F. Cristian, “Probabilistic Clock Synchronization,” Distributed Computing, vol. 3, no. 3, pp. 146-158, 1989.

[46] “Intel® 64 and IA-32 Architectures Software Developer’s Manual,” Volume 3B, System Programming Guide, Part 2 , 2010

[47] P. Domingos and G. Hulten, ”Mining high-speed data streams,” In Int’l Conf on Knowledge Discovery and Data Mining , (SIGKDD), pages 71–80, Boston, MA, 2000. ACM Press.

[48] J. Han and M. Kamber, “Data Mining: Concepts and Techniques,” 2nd ed., San Francisco: Elsevier, 2006.

[49] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, ”Mining data streams: a review,” SIGMOD Record, 34(2): 18–26, 2005.

[50] http://2011.hpcs.ca/

[51] http://www.sigops.org/osr.html

[52] http://www.igi-global.com/bookstore/titledetails.aspx?TitleId=1123