

A Comparison of Heaps and the TL Structure for the Simulation Event Set

W.R. Franta and Kurt Maly
University of Minnesota

Key Words and Phrases: simulation, event set, heaps, TL structure

CR Categories: 3.34, 4.22, 5.5, 8.1

Following publication of our paper [2], questions arose with respect to the superiority of the TL structure over heaps,¹ particularly in the face of the remarks of Gonnet [3], concerning the use of heaps for the physical realization of the simulation event set. Gonnet's communication was in response to the Vaucher and Duval paper [5], and suggested the heap to be a more efficient structure than any proposed in [5]. As regards a comparison of heaps and the TL structure we can make the following remarks:

1. For the TL structure consecutive elements can be identified in $O(1)$, for heaps $O(n)$ (n denotes the number of notices in the set). Such identification is necessary to

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

Authors' address: Department of Computer Science, University of Minnesota, Minneapolis, MN 55455.

¹ We are indebted to Dennis Kibler, who suggested in a letter that this comparison be made.

© 1978 ACM 0001-0782/78/1000-0873 \$00.75

support the scheduling constructs found in some extant languages, for example, see [1].

2. With the TL structure a FIFO or LIFO ordering is maintained automatically. For the heap a secondary field giving the event notice insertion time is necessary.

3. Simple scheduling when the distribution of notices in the event set is uniform is $O(1)$ for the heap, and $O(1)$ for the TL structure (experimentally determined) and with the TL structure is nearly $O(1)$ for all distributions tested, see [2].

4. The average and worst case complexity for the hold (\cdot) operation (see [2] for a description) is $O(\log n)$ for heaps while average complexity for the TL structure is $O(1)$, experimentally determined, and the worst case complexity is $O(\sqrt{n})$. We consider this significant as hold (\cdot) is a frequently encountered operation. Additionally note that normal event processing or the cancel operation (necessary for interruptible activities) are on the average $O(1)$ with TL and $O(\log n)$ for heaps.

5. To store the event nodes heap requires $\frac{1}{3}$ of the storage needed for the TL structure. If the heap maintains FIFO ordering of notices the ratio becomes $\frac{2}{3}$. Note, however, that for either approach the storage requirement for as many as 500 notices is of the order of 1K storage cells.

In most situations the critical factor is execution time and not the storage required. We believe, therefore, that while the above comparisons are important, the most significant comparison is algorithm performance in a variety of generally encountered situations. To gain insight into the expected performance, and allow comparison with the expected performance of the TL structure we performed the same series of experiments for the heap as reported for the TL structure (see [2] for a description of the experiment structure). Both algorithms were tested using Pascal, were run on the Cyber 74, were implemented to insure a fair comparison, and are available from the authors. A summary of our findings is given in Figures 1-3.

In Figure 1 the average execution time per hold operation is reported for the heap maintaining FIFO ordering. In [2, 5] the hold operation was selected for testing as it required both a remove and schedule operation. When using the heap those operations can be combined into a single operation, say "sift-up." One set of curves reflects the average hold times with remove and schedule separate, the other with them combined. That is, in one case hold is essentially realized as

remove (notice);
schedule (notice);

and in the other as

sift-up (notice);

The two lower curves reflect the envelope of hold times for the TL structure, see Figure 5 in [2]. Figure 2 gives similar results when FIFO ordering in the heap is not

Fig. 1. Comparison of Heap Maintaining FIFO Ordering and TL Structure.

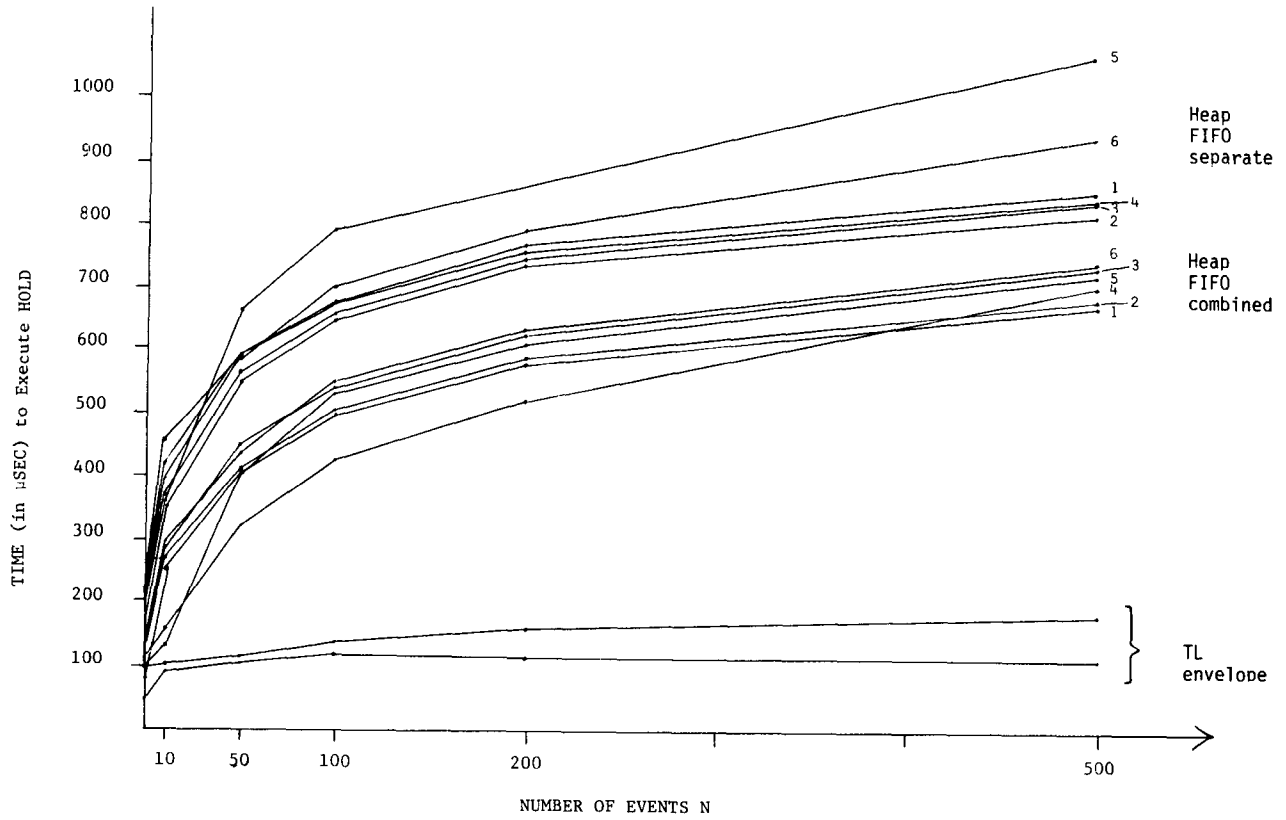


Fig. 2. Comparison of Heap Not Maintaining FIFO Ordering and TL Structure.

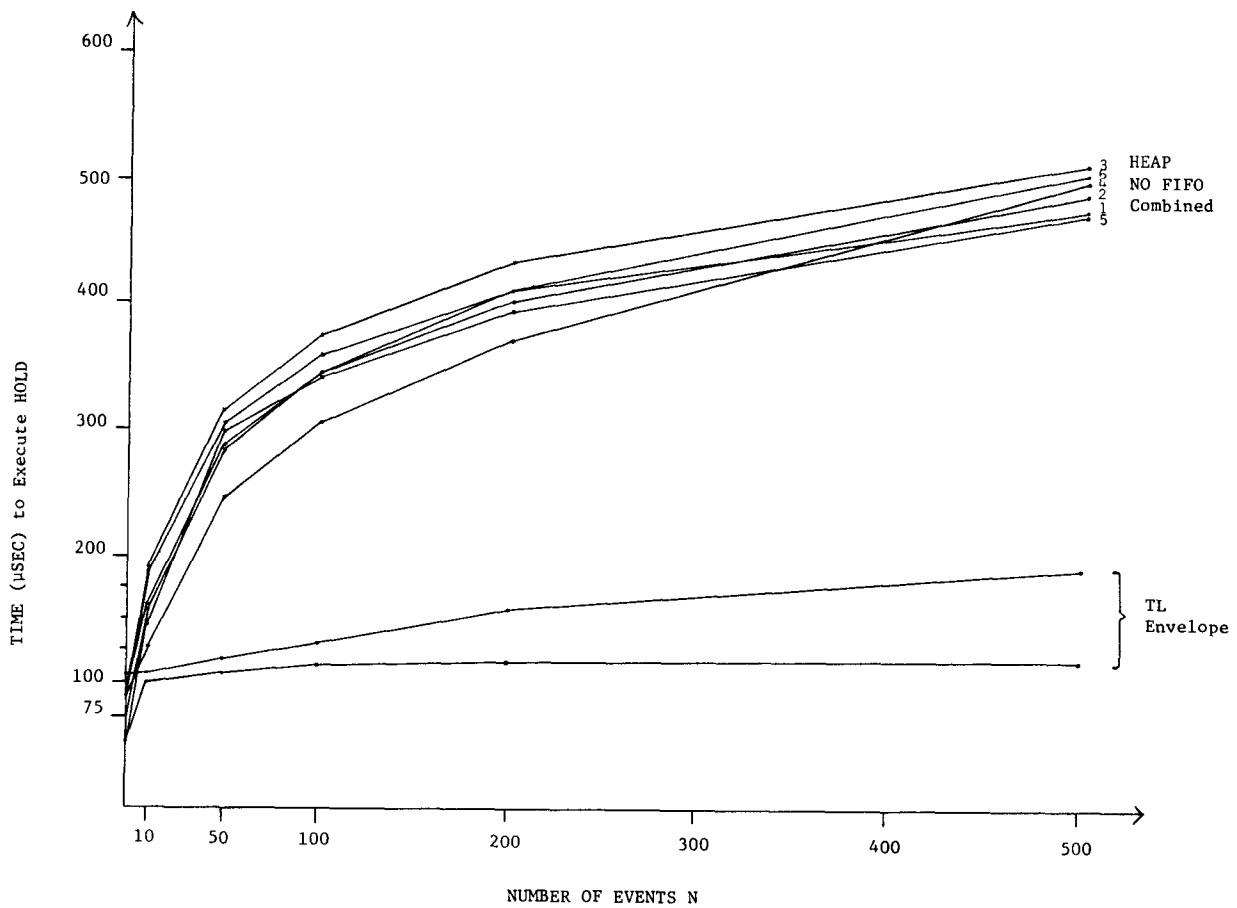
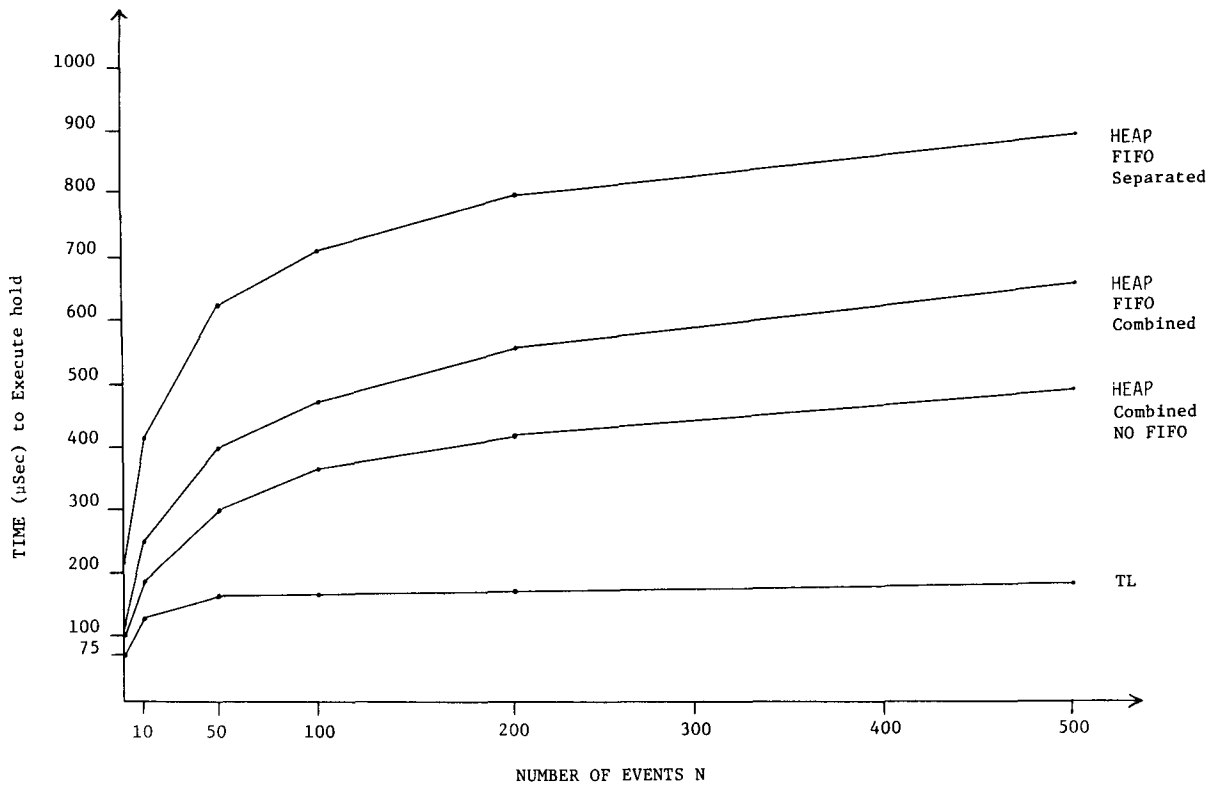


Fig. 3. Mixed Distribution Test Results.



maintained. Figure 3 shows results for a "mixed" distribution, see Figure 6 in [2] for details.

In all cases we observed the expected logarithmic behavior for the heap. That is, while the graphs imply an $O(1)$ complexity for the schedule operation, independent of the distribution of times in the event set, the equally frequent remove operation is $O(\log n)$. Further they reflect the overhead associated with maintaining FIFO ordering. On the basis of points 1-5 above and Figures 1-3, we must conclude that the TL structure is to be preferred over the heap.

In closing, we note that we made comparisons via experimentation rather than analysis as information on the distribution of times in the event set is a difficult

problem, and only recently has progress been made in describing those distributions, see [4, 6].

Received November 1977

References

1. Franta, W.R. *The Process View of Simulation*. Elsevier North-Holland, 1977.
2. Franta, W.R., and Maly, K. An efficient data structure for the simulation event set. *Comm. ACM* 20, 8 (Aug. 1977), 596-602.
3. Gonnet, G.H. Heaps applied to event driven mechanisms. *Comm. ACM* 19, 7 (July 1976), 417-418.
4. Jonassen, A., and Dahl, O.-J. Analysis of an algorithm for priority queue administration. *BIT* 15 (1975), 409-422.
5. Vaucher, J.G., and Duval, P. A comparison of simulation event list algorithms. *Comm. ACM* 4, 18 (April 1975), 223-230.
6. Vaucher, J.G. On the distribution of event times for the notices in a simulation event list. *INFOR* 15, 2 (June 1977), 171-182.