PERFORMANCE OF FUTURE EVENT SET IMPLEMENTATIONS

G. LENCSE

DEPARTMENT OF TELECOMMUNICATIONS
TECHNICAL UNIVERSITY OF BUDAPEST
E-MAIL: GABOR.LENCSE@HIT.BME.HU

Seven data structures were examined with simulation and compared to find out which one should be used to implement the Future Event Set of an event driven discrete event simulator. The number of key comparisons and pointer references as well as the CPU time were measured. The effect of the different CPU architectures was examined. Various parameter settings were used to determine the characteristics of the studied event set implementations.

1. INTRODUCTION

In an event-driven discrete event simulator the events (the state changes of the system) are stored in the Future Event Set (FES). The data structure and the algorithms used for storing the elements of the FES influence the speed of the simulator to a great extent.

The most common data structures have already been studied and their general performance characteristics have been determined. Most investigations assume that the keys are random (with uniform distribution), the number of search operations is much higher than the number of insertions or deletions and the element to be searched or deleted is also randomly chosen. Nevertheless, in the case of the FES, the keys are not uniformly distributed, but show a rising tendency; in the vast majority of cases, the first element is deleted and a new one is inserted. Sometimes randomly chosen elements are deleted, but no other search operations are used.

Reeves [1] examined variants of lists and heaps, but the different kinds of tree structures and the skip list [2] required further study.

The examined data structures were: ordered single-linked list, binary tree, AVL-tree, B-tree, 2-3-tree, heap and skip list.

2. THE INVESTIGATION

The behavior of a discrete-event simulator (from the viewpoint of the Future Event Set) was simulated in the following way: *new* events were inserted into the FES, then the *first* event (the one with the smallest time stamp) was taken out or a randomly chosen event was deleted. This procedure was repeated many times and the time stamp of the *new* event was the sum of the time stamp of the most recently removed "*first*" event and a random delay computed according to different distributions. The parameters were: the number of the events in the FES, the state of the FES (transient or steady), the proportion of the randomly deleted events and the distribution of the delay (exponential, uniform, and two normal distributions with different deviations).

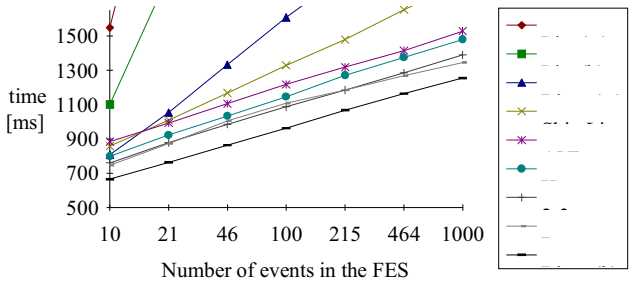

Fig. 1. Execution time of 1000 simulation steps, i486 DLC proc.

To determine the performance characteristics of the studied event set algorithms, the number of key comparisons and of pointer references as well as the CPU time were measured. By changing the parameters of the simulation model orthogonally, simulations were run with all possible parameter combinations. The simulation was performed on the following processors: Intel 486DLC, Intel 486DX, DEC ALPHA.

3. RESULTS

By examining the number of key comparisons, it was found that the balanced trees and heap produce the lowest values among all the data structures. For this reason, if we use a processor where the CPU time of the event set operations is dominated by the key comparisons (that is, floating point operations are not supported by hardware), balanced trees and heap are the best choice. Binary tree produced good performance characteristics in the case of exponential distribution, though in the case of other distributions binary tree showed worse results. (Fig. 1. b=best, w=worst case)

On the other hand, skip list produced much better time characteristics than the balanced trees when advanced hardware floating point processing was used. (Fig. 2.)

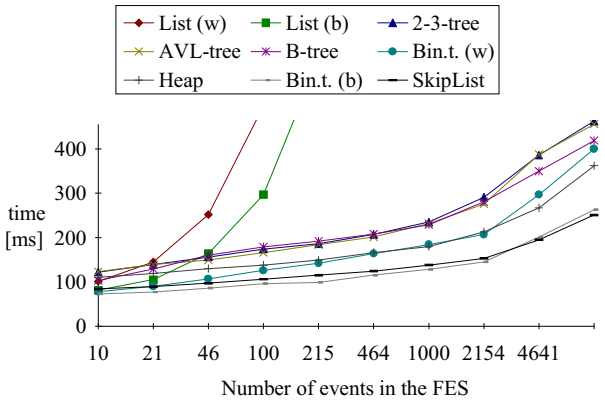

Fig. 2. Execution time of 10000 simulation steps, DEC ALPHA proc.

With strong floating point support, it is worth using skip list rather than balanced trees, especially because its algorithms are even simpler than that of the balanced trees.

4. CONCLUSION

We concluded that both the processor type and the distribution of the delay of the new events influence which data structure produces the best time properties. Though balanced trees proved to be quite good, heap and especially skip list was found to be significantly better on modern processor architectures.

REFERENCES

[1] Reeves, C. M. 1984. "Complexity Analyses of Event Set Algorithms" *The computer journal* 27. no. 1, 72-79

[2] Pugh, V. 1990. "Skip Lists: A Probabilistic Alternative to Balanced Trees" *Commun. of the ACM* 33, no. 6, 668-676

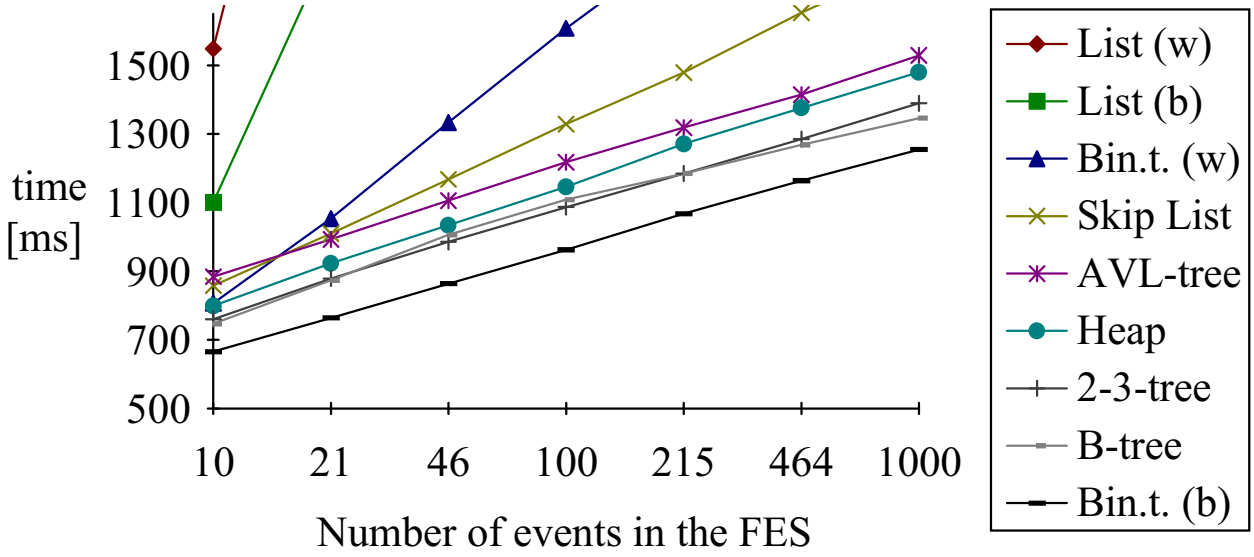

Fig. 1. Execution time of 1000 simulation steps, i486 DLC proc.

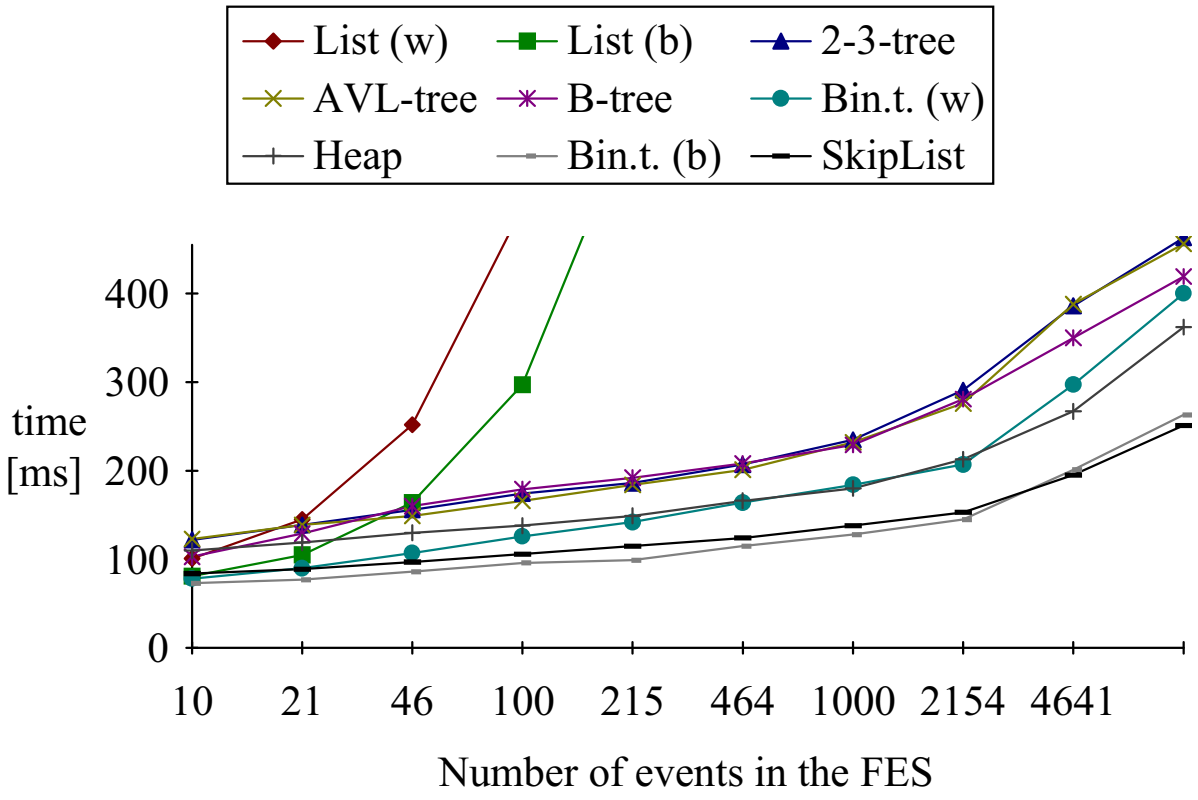

Fig. 2. Execution time of 10000 simulation steps, DEC ALPHA proc.

Fig. 1. and Fig. 2. in doubled size for the article:
Gábor Lencse: PERFORMANCE OF FUTURE EVENT SET IMPLEMENTATIONS

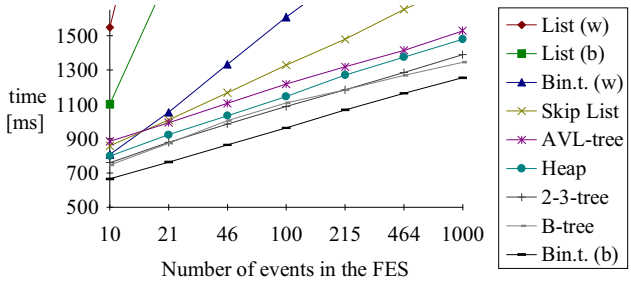

Fig. 1. Execution time of 1000 simulation steps, i486 DLC proc.