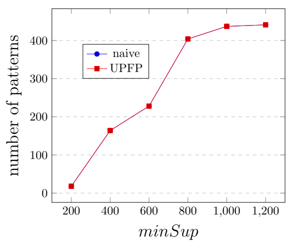
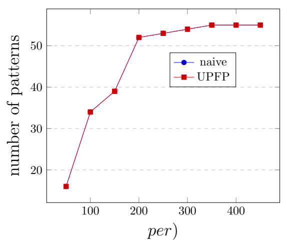
1 Additional experiments

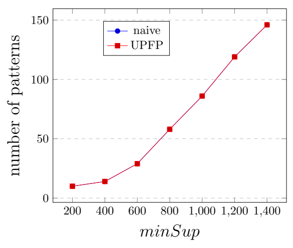
Due to page limitation, we reported experiments conducted by varying $minSup$ value in main paper.Here we are reporting additional experiments by varying $maxPer$ value. The $minSup$ in T10I10D200K, Congestion and Retail databases have been set at 800, 220, and 200, respectively.



(a) T10I10D200K



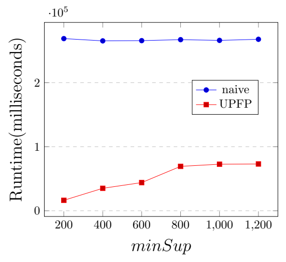
(b) Congestion



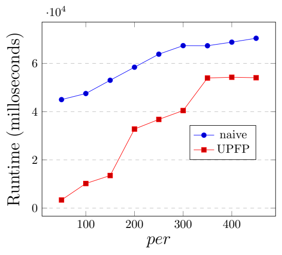
(c) Retail

Figure 1: Periodic-frequent patterns generated in various databases

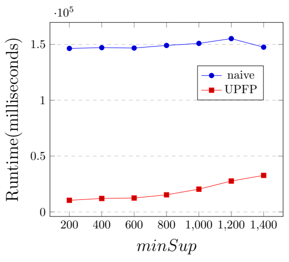
Figures 1a, 1b and 1c show the number of periodic-frequent patterns generated at different $maxPer$ values in T10I10D200K, Congestion and Retail databases, respectively. It can be observed that increase in $maxPer$ increases the number of periodic-frequent patterns being generated. It is because many patterns appear in long time to satisfy the increased $maxPer$ value.



(a) T10I10D200K



(b) Congestion

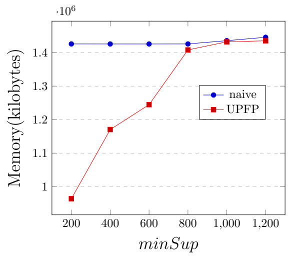


(c) Retail

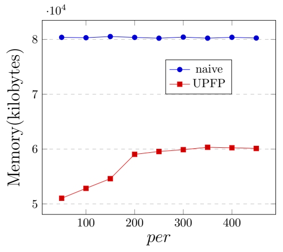
Figure 2: Runtime requirements of naïve and UPFP-growth algorithms

Figures 2a, 2b and 2c show the runtime requirements of UPFP-growth and naïve algorithms at different $maxPer$ values in T10I10D200K, Congestion and Retail databases, respectively. The following two observations can be drawn from these figures: (i) increase in $maxPer$ increases the runtime requirements of both the algorithms. It is because both algorithms have to discover many periodic-frequent patterns. (ii) UPFP-growth outperforms naïve algorithm on every database at any given $maxPer$ value. The runtime gap between both the algorithms increases with the increase in $maxPer$ value.

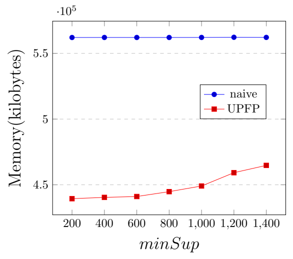
Figures 4a, 4b and 4c show the memory requirements of UPFP-growth and naïve algorithms at different $maxPer$ in T10I10D200K, Congestion and Retail



(a) T10I10D200K



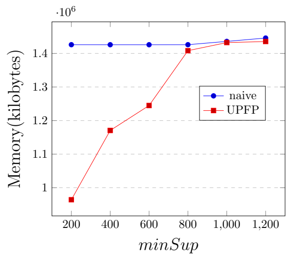
(b) Congestion



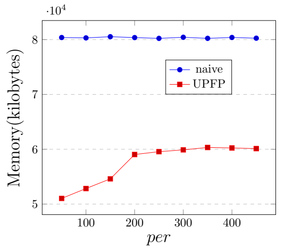
(c) Retail

Figure 3: Memory requirements of naïve and UPFP-growth algorithms

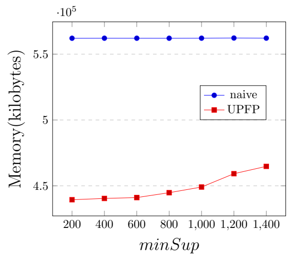
databases, respectively. Similar statements as stated above can be said regarding the memory consumption of both algorithms. More important, it can be observed that memory requirements of UPFP-growth were order of magnitude times lower than the naïve algorithm on real-world databases.



(a) T10I10D200K



(b) Congestion

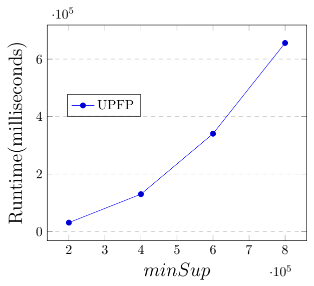


(c) Retail

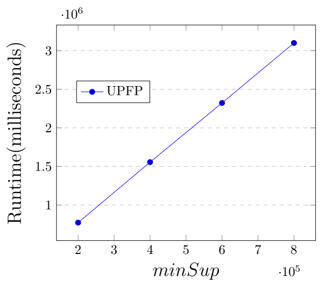
Figure 4: Memory requirements of naïve and UPFP-growth algorithms

1.1 Scalability test

The Kosarak database was divided into five portions of 0.2 million transactions in each part. Then we investigated the performance of UPFP algorithm after accumulating each portion with previous parts. Fig. 5a and 5b, respectively show the runtime and memory requirements of UPFP algorithm at different database sizes when $minSup = 1000$ (in count) and $maxPer = 100$ (in count). The following two observations can be drawn from these figures: (i) runtime, and memory requirements of UPFP algorithm increase almost linearly with the increase in database size.



(a) Execution Time



(b) Memory

Figure 5: Scalability of PFP-growth++ and CPFP-Miner