Randomized Methods in Computational Geometry

(Marco Pellegrini)

Lecture 2: Skip Lists

1

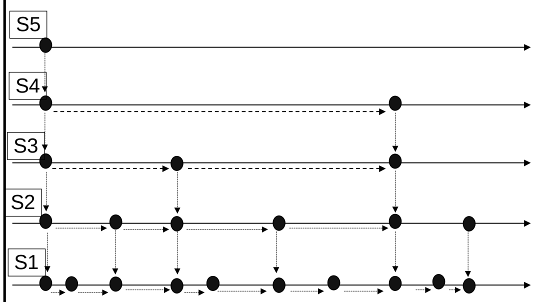
Skip lists

(dynamic searching in sorted lists)

We examine a data structure (Skip lists) for searching that allow efficient insertion/deletion of elements.
Let S be a set of n points on the real line R. We build a sequence of r sets S_1, S_2,... S_r such that S_1=S, S_r is empty and S_{i+1} is a subset of S_i.
To do so we flip a fair coin for each element of S_i, If the outcome is head the point is promoted to S_{i+1}.
The skip list is formed by H(S_1),...,H(S_r), plus a link between copies of the same point in the next set.

2

Skip lists, example.



3

Searching algorithm

- Let q be the query point. We locate by sequential scan of q in H(S_r). Assume we have located q in A in H(S_i), we continue the search in the intervals of H(S_{i-1}) which are contained in A.
- Probabilistic analysis: The number of levels r is O(log n) with high probability.
A single point is in k levels with probability 2^{-k}, thus the probability that at least one is in k levels is less than $n2^{-k}$. For k=c log n this probability is polynomially small.
- A node is in i levels with probability 2^{-i}, thus the expected value is:

$$\sum_{i \geq 1} \frac{i}{2^i} = O(1)$$

4

Analysis, cntd.

Thus the expected storage is O(n). Using Chernoff technique we can show it holds with high probability.
The skip list is constructed in expected time O(m log m).
Now we estimate the expected search time, with respect to the random choices of the building algorithm. Let q be a query point and for each level i we denote with x(q,i) the number of points to the left of q in the interval of H_i containing q. These are the points visited during the search. The number of points in H_i follows the geometric distribution, therefore E[x(q,i)] = O(1). To conclude the analysis see the notes.

5

Insertions and deletions

- To insert a point q in we first search for q then we update H(S_1) with q, we flip a fair coin until we get a failure. If we have t < r successes we add q to H(S_2) ... H(S_t). If j > r we add extra levels for r+1,..,t.
- The expected time is bounded by the search, thus O(log n).
- To delete a point q, we just delete it from all levels. We search for it first so the time is still bounded by the search: O(log n).

6