after sorting according to width in increasing order (if two envelopes have the same width, sorting according to height in decreasing order). then we only need to compute LIS in 1D (with strict $<$).
there's an $O(n \log \log n)$ time algorithm [3] using vEB trees, and also $O(n \log \log \text{ans})$ [1] for computing LIS of a permutation with n numbers, and it's easy to wlog assume the heights are distinct.
computing LIS needs $\Theta(n \log n)$ time in the comparison based model [2].

References

[1] Maxime Crochemore and Ely Porat. Fast computation of a longest increasing subsequence and application. *Information and computation*, 208(9):1054–1059, 2010.

[2] Michael L Fredman. On computing the length of longest increasing subsequences. *Discrete Mathematics*, 11(1):29–35, 1975.

[3] James W Hunt and Thomas G Szymanski. A fast algorithm for computing longest common subsequences. *Communications of the ACM*, 20(5):350–353, 1977.