Overmars and van Leeuwen(1981) Algorithm

For online convex hull computation

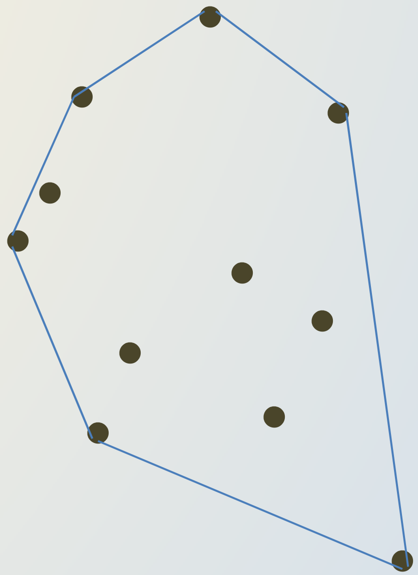
Michal Fuksa
fuksamic@fel.cvut.cz

Contents

- Dynamic convex hull
- Algorithmic approach
- Data structure
- Challenges
- Example
- Conclusion
- Questions?

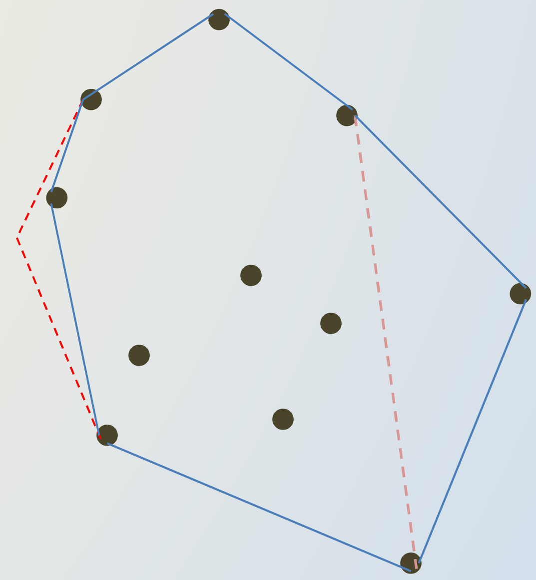
Convex hull

- Smallest convex set, which is superset of given set.
- Convex set: Set of points: $\forall u,v \in A$, $k \in \langle 0,1 \rangle$: $u.k+v(1-k) \in A$,
- We are looking for boundary



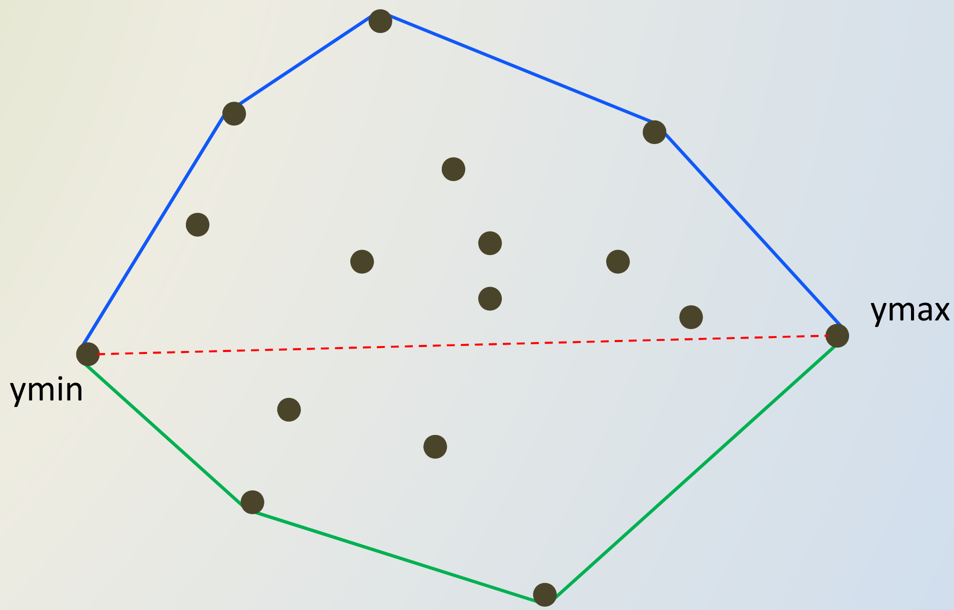
Algorithm properties

- Online construction
- Dynamic convex hull
- Point removal – must store all the points in convenient way
  - Could be done by reconstructing whole structure from scratch (expensive)
- 2D only
- Query in O(lg(n)), Update in O(lg^2(n))



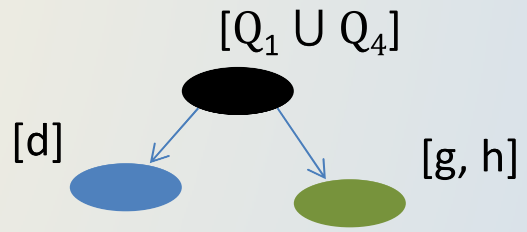
Construction

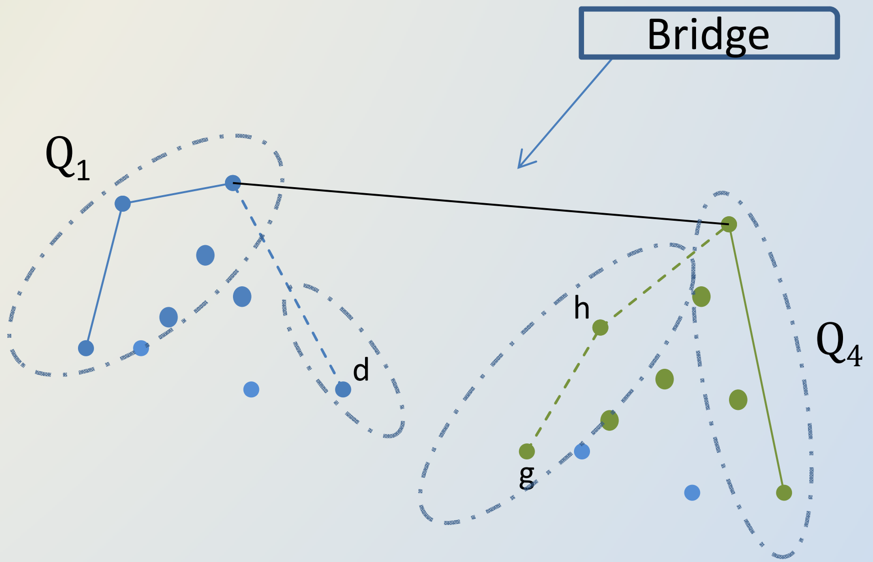
- Solve upper(UH) and lower(LH) convex hull separately
- Final CH is union of UH and LH
- Construction is symmetric



Data structure – Binary tree

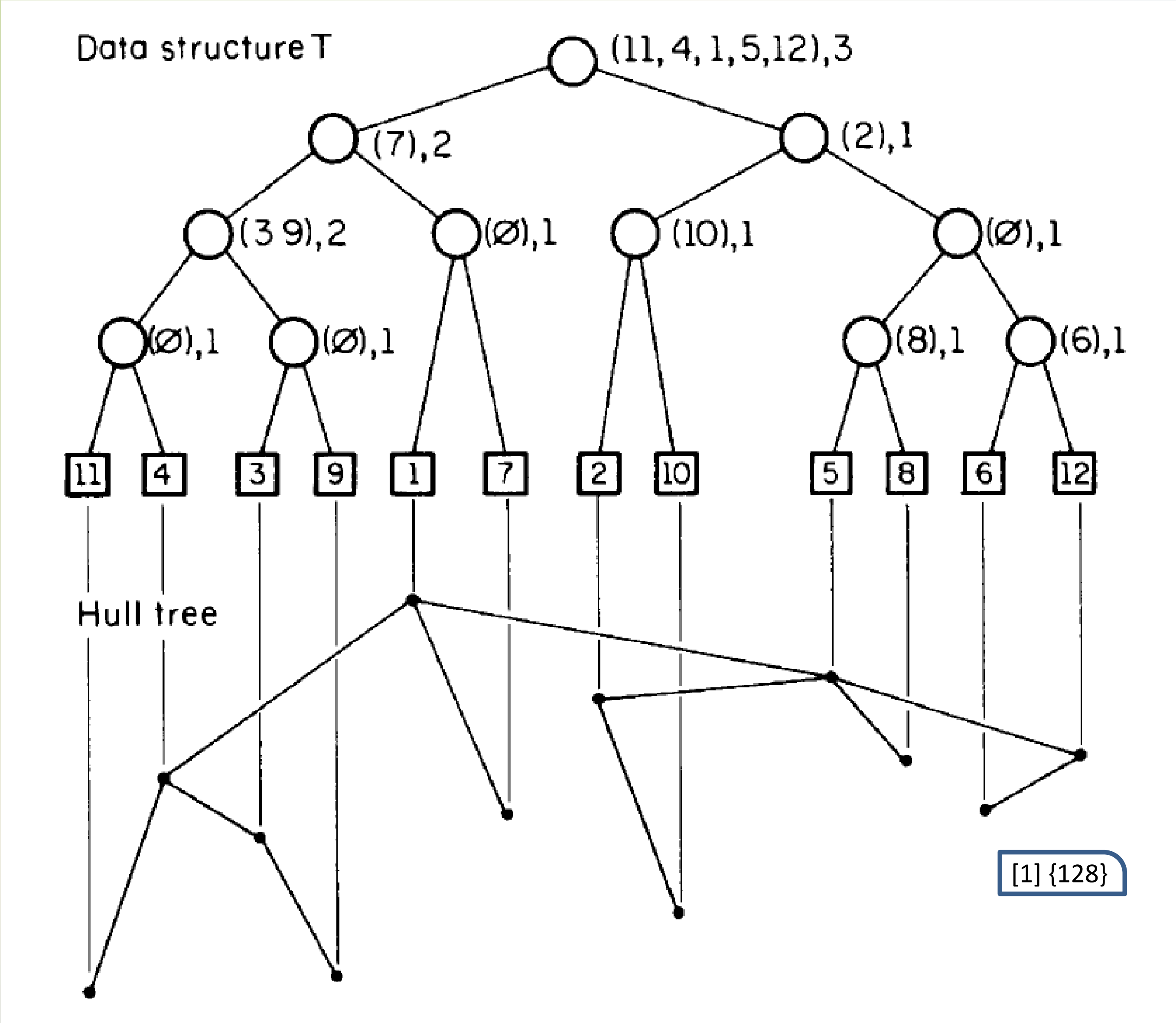
- Each interior node of the tree represents UH
- Only leaves are points





- Each node keeps information about points of CH, which are **not** in parent CH.
- **<u>Concatenable queue</u>**(also tree, insert,delete,find,concat,cut in lg(n))
- Queue in root contains all points of upper(lower) convex hull

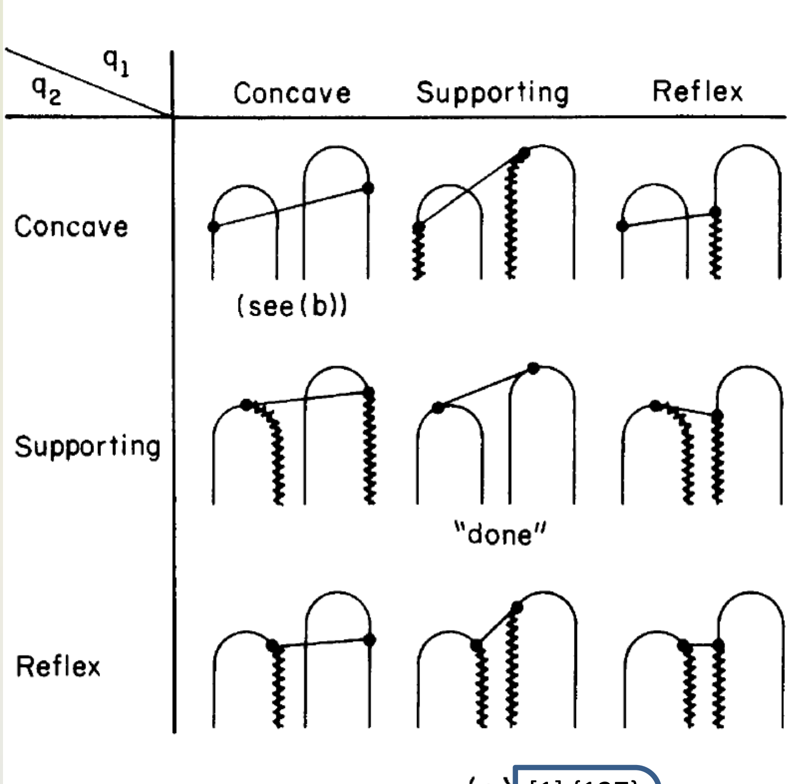
Data structure – Binary tree



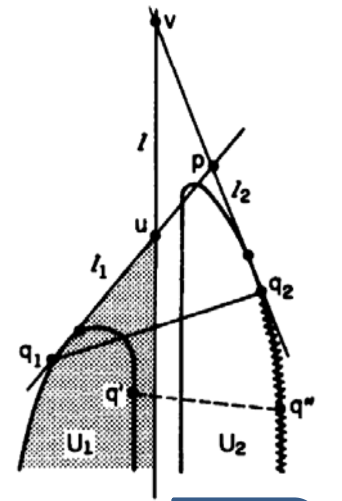
[1] {128}

Construction - Bridging

- Bridging problem
- From two distinct CH create one.
- Apply binary search on both CH



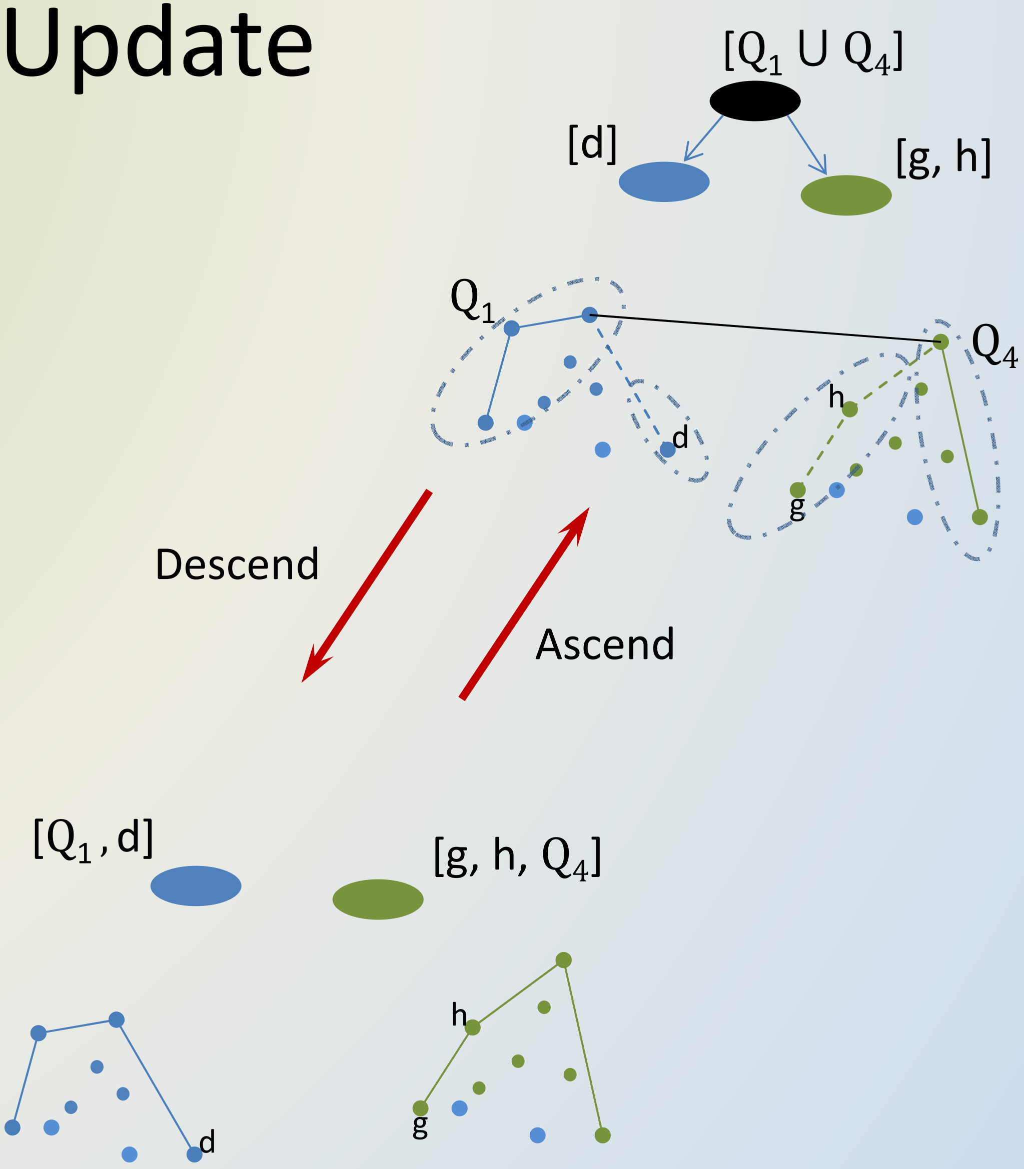
(a) [1] {127}



(b) [1] {127}

Update

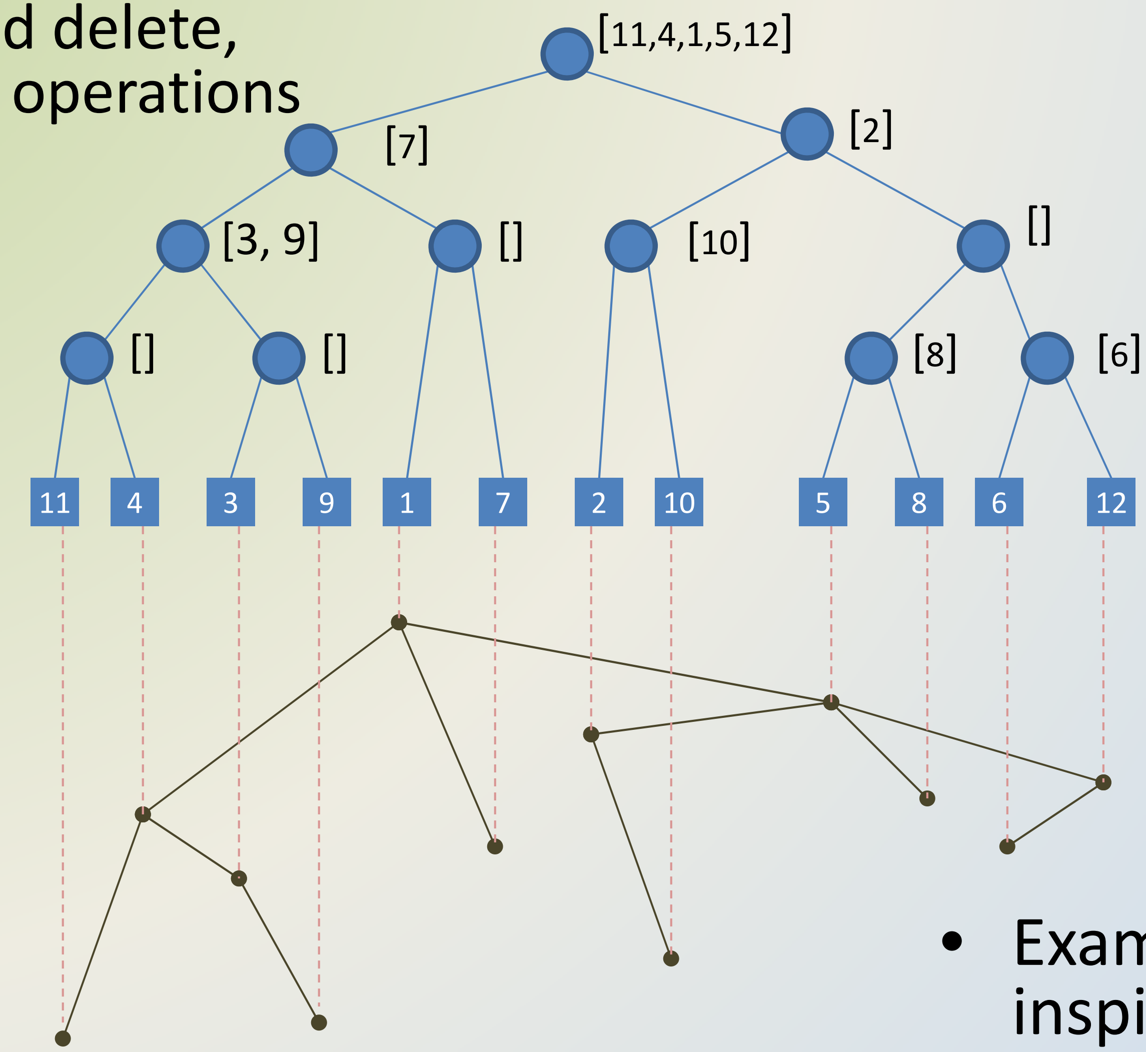
- Add and delete, Similar operations
- Algorithm:
  - Descend (Splitting)
  - Update
  - Ascend (reconstruct)



Nope

Update

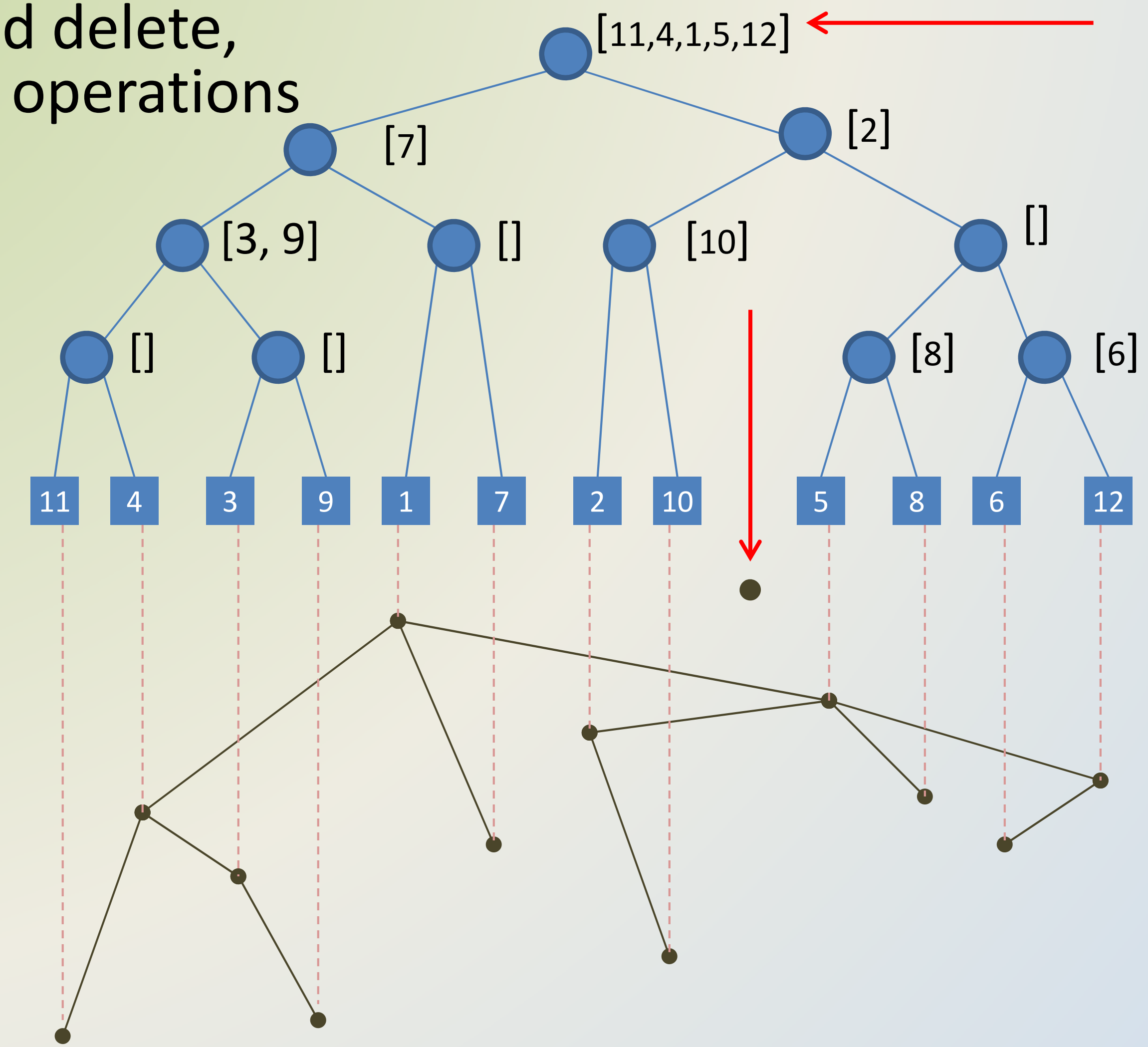
- Add and delete,
  Similar operations



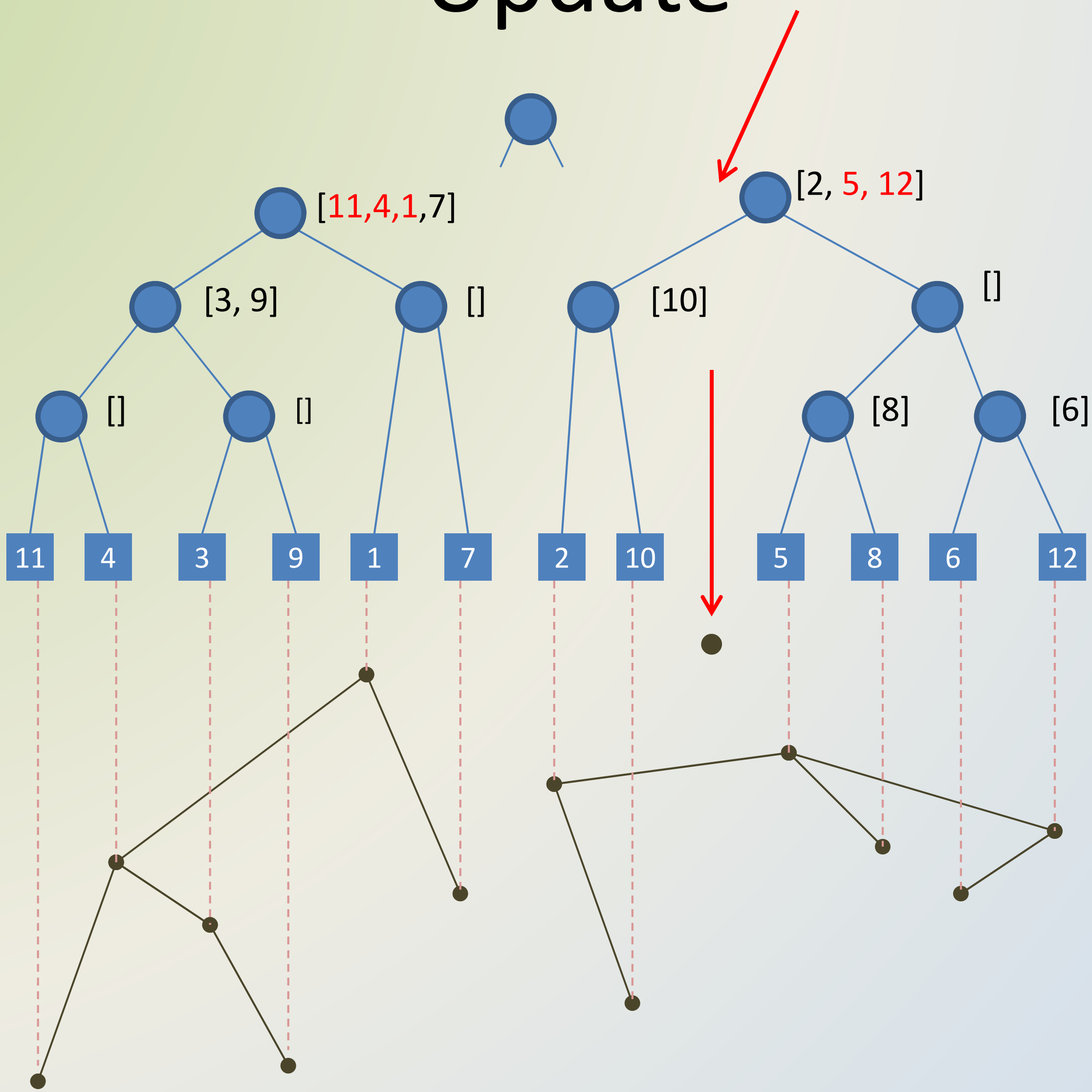
- Example inspired by [1] {131}

Update

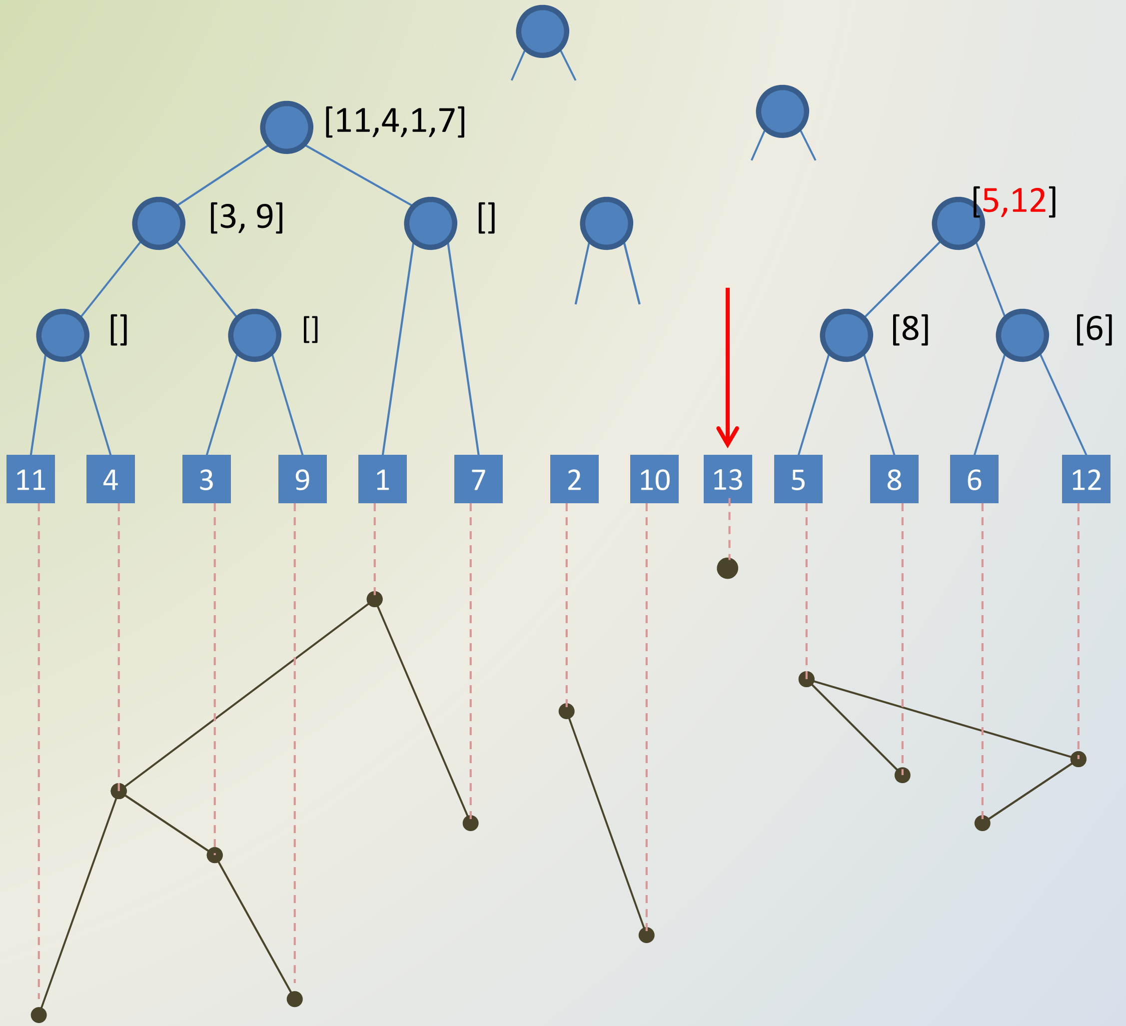
- Add and delete, Similar operations



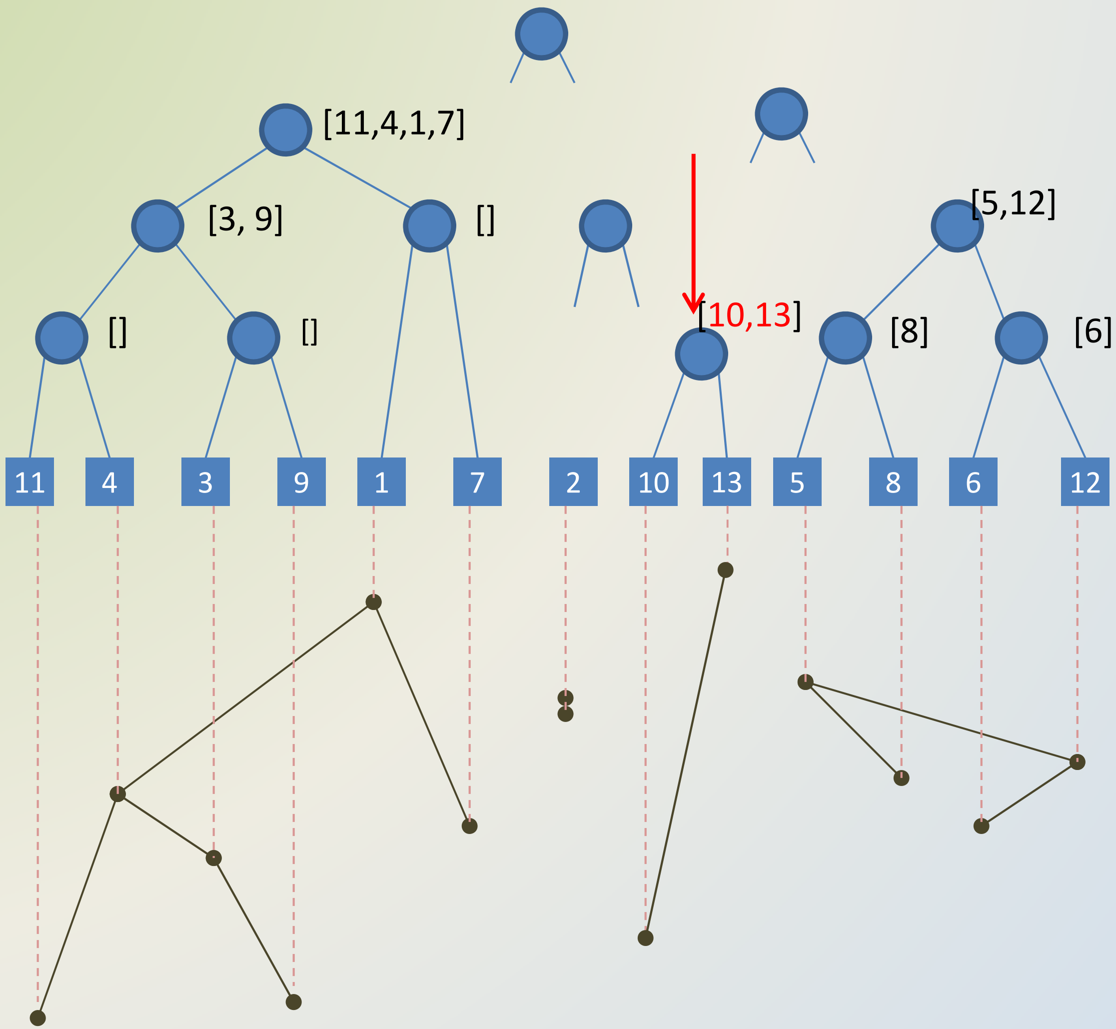
Update



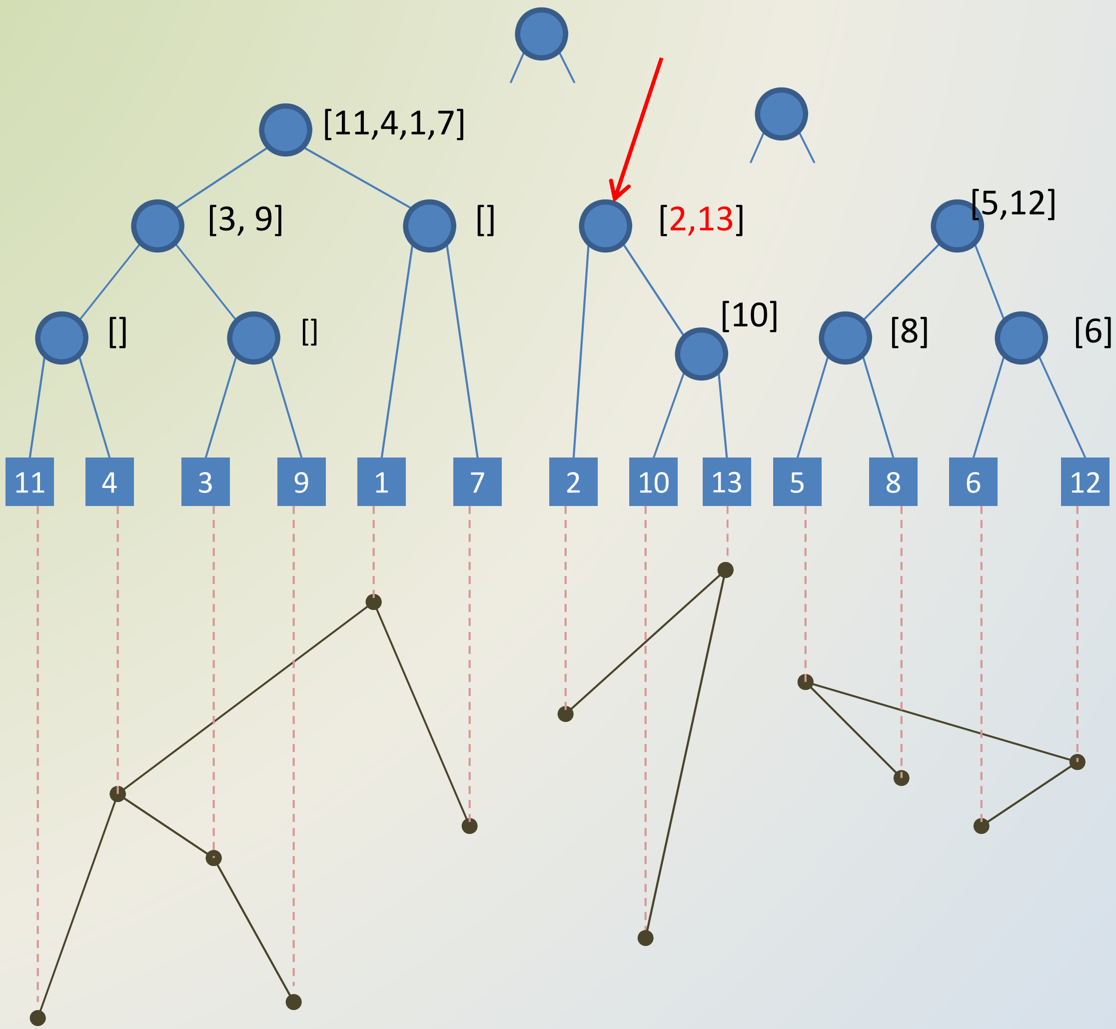
Update



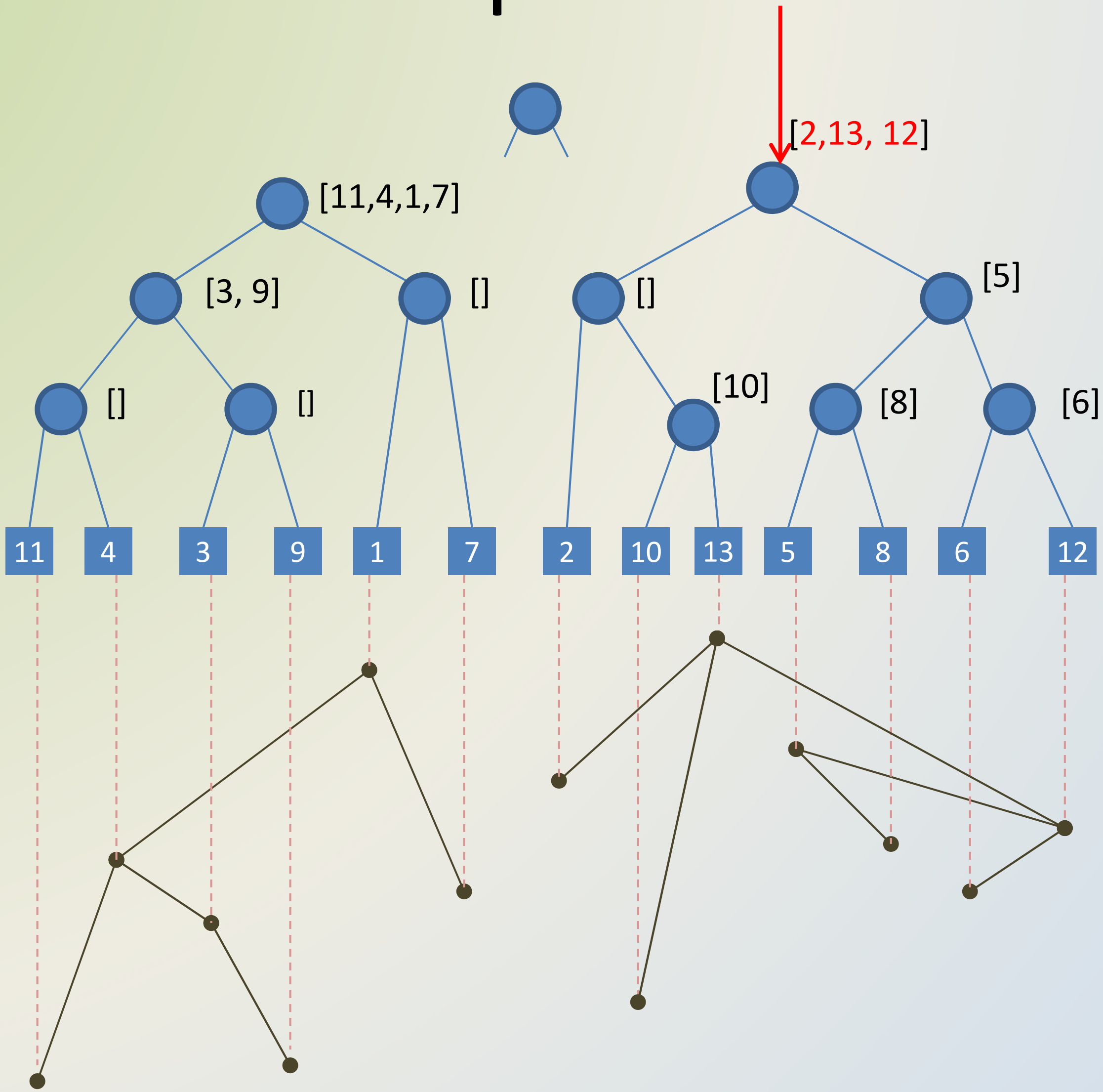
Update



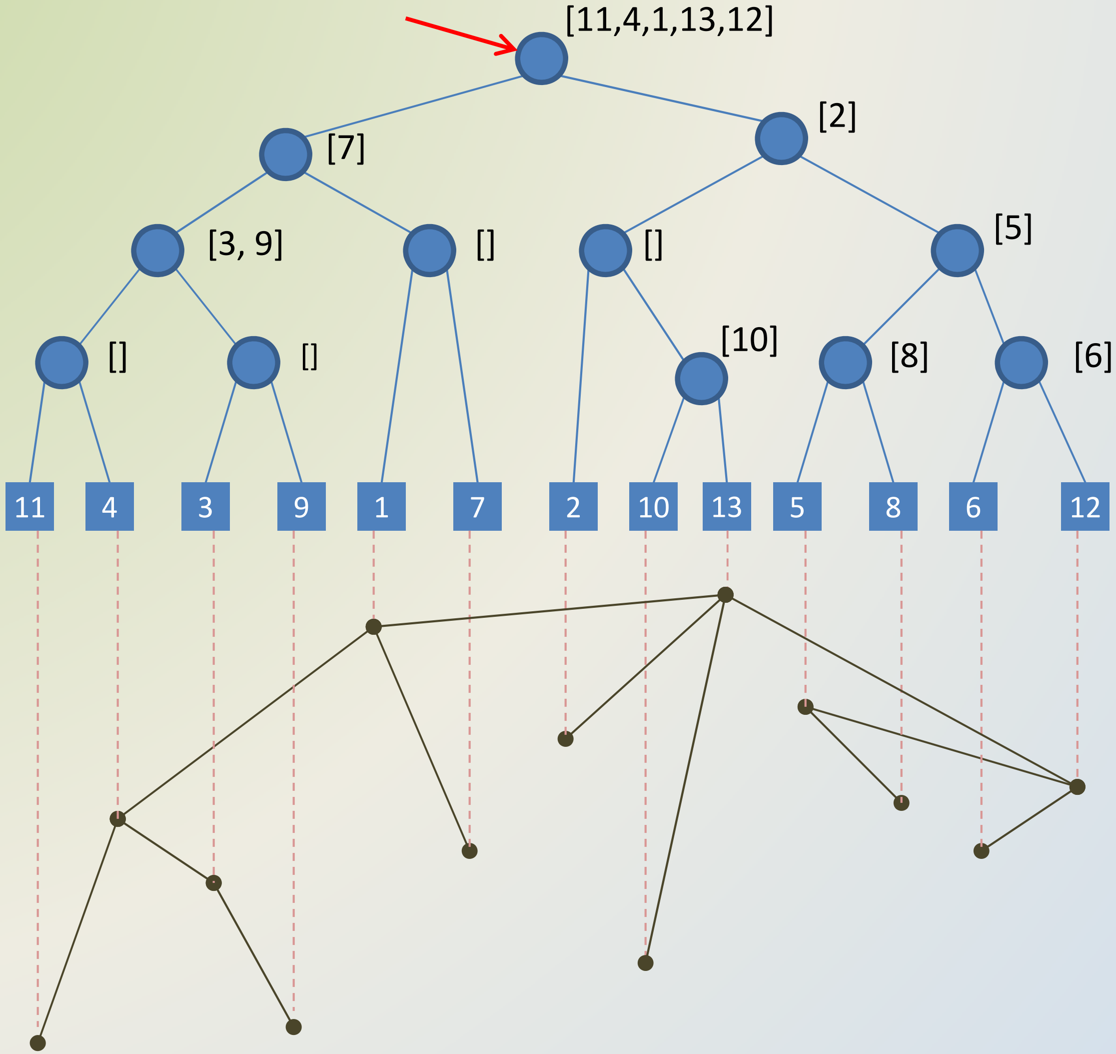
Update



Update



Update



Done

Complexity

- Add and delete are almost the same
- Descending: log(n)
- Update: 1
- Ascend and reconstruction: log(n)* log(k)
  - Ascend + Bridging, k<n
- Query: O(log(n))
- Update: $O(\log^2(n))$
- Build: $O(n. \log^2(n))$!
- Space: O(n)

- Unsurpassed until T. M. Chan. Dynamic planar convex hull operations in near-logarithmic amortized time($\log^{(1+\varepsilon)}(n)$) – after 20 years
- Current optimal algorithm have log(n) update time.(Brodal,Jacob), [3]

Pseudo-code

```
    procedure DESCEND(v, p)
begin if (v ≠ leaf) then
        begin (Q_L, Q_R) := SPLIT(U[v]; J[v])
              U[LSON[v]] := SPLICE(Q_L, Q[LSON[v]]);
              U[RSON[v]] := SPLICE(Q[RSON[v]], Q_R);
              if (x(p) ≤ x[v]) then v := LSON[v] else v := RSON[v];
              DESCEND(v, p)
        end
end.
```

```
    procedure ASCEND(v)
begin if (v ≠ root) then
        begin (Q_1, Q_2, Q_3, Q_4; J) := BRIDGE(U[v], U[SIBLING[v]]);
              Q[LSON[FATHER[v]]] := Q_2;
              Q[RSON[FATHER[v]]] := Q_3;
              U[FATHER[v]] := SPLICE(Q_1, Q_4);
              J[FATHER[v]] := J;
              ASCEND(FATHER[v])
        end;
      else Q[v] := U[v]
end.
```

[1] {131}

References

- [1] Computational Geometry, An Introduction: Franco P. Preparata, Michael Ian Shamos {1985}
- [2] Time-Space Optimal Convex Hull Algorithms, Hla Min, S. Q. Zheng
- [3] Dynamic Planar Convex Hull, Gerth Stølting Brodal, Riko Jacob