Efficient and probabilistic heavy ion jet reconstruction and complexity constraints (Journal club August 9, 2006)

Yue Shi Lai

Topics

- Review: “Classical” k_T algorithm
- Review: Balanced trees
- “Classical” k_T is not $O(N^3)$
- k_T in $O(N^2)$
- Review: Voronoi diagrams
- k_T in $O(N \log N)$
- Extension to probabilistic algorithms

Review: “classical” k_T algorithm

- Classical k_T algorithm are thought to require $O(N^3)$ operations:
  - Initial (d_{ij}) table construction . . . . $O(N^2)$
  - N steps, for each:
    - Find $\min(d_{ij}, d_{iB})$ (exhaustive search) . . . . $O(N^2)$
    - Remove protojet (array deletion by swapping) . . . . $O(1)$
    - Recalculate d_{ij} (array replacement) . . . . $O(N)$
- Is that really so?

Balanced tree I

- height of the tree (G. M. Adelson-Velsky & E. M. Landis, 1962):

$$\log_2(N+1) \leq h \leq 1.4405 \log_2(N+2) - 0.3327$$

- In the functional notation (Pseudo-Haskell):
  - `data Tree t = Null | Bin (Tree t) t (Tree t)`
  - $\alpha^{(h)}$ denotes a subtree with height h
- Elementary transformations:

  `single_rotation (Bin` $\alpha^{(h)}$ `A (Bin` $\beta^{(h)}$ `B` $\gamma^{(h+1)}$`)) =`
  `Bin (Bin` $\alpha^{(h)}$ `A` $\beta^{(h)}$`) B` $\gamma^{(h+1)}$

  `double_rotation (Bin` $\alpha^{(h)}$ `A (Bin (Bin` $\beta^{(h-1)}$ `X` $\gamma^{(h)}$`) B` $\delta^{(h)}$`)) =`
  `Bin (Bin` $\alpha^{(h)}$ `A` $\beta^{(h-1)}$`) X (Bin` $\gamma^{(h)}$ `B` $\delta^{(h)}$`)`

  and their reflections (for left inbalance).

Balanced tree II

- Each rotation has $O(1)$ complexity.
- $O(\log N)$ (but possibly $O(k)$) asymptotic complexity for:
  - Find an item with a specified key . . . . $O(\log N)$
  - Find the kth item . . . . $O(k \log N)$
  - Insert an item at a specified place (find and $O(1)$ rotation) . . $O(\log N)$
  - Delete an specified item (find and $O(\log N)$ rotations) . . . . $O(\log N)$

Complexity of k_T algorithm II

- What happened, had S. D. Ellis & D. E. Soper (1993) and many others implement it without much deeper thought, but "right"?
  - Initial (d_{ij}) table construction . . . . $O(N^2)$
  - N steps, for each:
    - Find $\min(d_{ij}, d_{iB})$ (obtain a balanced tree element) . . . . . . $O(\log N)$
    - Remove protojet (balanced tree deletion) . . . . $O(N \log N)$
    - Recalculate d_{ij} (balanced tree insertion) . . . . $O(N \log N)$
- We have an $O(N^2 \log N)$ k_T algorithm without doing anything and for free!
- $O(N^3)$ is just because of choosing the wrong data structure.

FastJet I

M. Cacciari & G. P. Salam (1995) claims out:

- k_T algorithm is $O(N^2)$ if a back-referenced list of $\min(d_{ij}, d_{iB})$ is used.
- k_T algorithm is $O(N \log N)$ if incremental Delaunay triangulation is used.

FastJet works for a certain class of multiplicative weights:

- $w_{ij} = \min(k_{T,i}, k_{T,j})$
- $d_{ij} = w_{ij} R_{ij} \quad \Rightarrow \quad R_{ij} = \min!$
- $O(1)$ nearest neighbours are exhaustively searched.

FastJet II

- G being the geometrical nearest neighbour
  - Initial (d_{iG}) table construction . . . . $O(N^2)$
  - N steps, for each:
    - Find $\min(d_{iG}, d_{iB})$ (obtain a balanced tree element) . . . . . . $O(\log N)$
    - Remove protojet (balanced tree deletion) . . . . $O(\log N)$
    - Recalculate d_{iG} (balanced tree insertion) . . . . $O(\log N)$
- Note the number of updates for d_{iG} per step is bounded by 6.

Voronoi diagram I

- The Voronoi regions are:

$$V(p) = \{x : |p - x| \leq |q - x|, \forall q \in S\}$$

- The Voronoi diagram is:

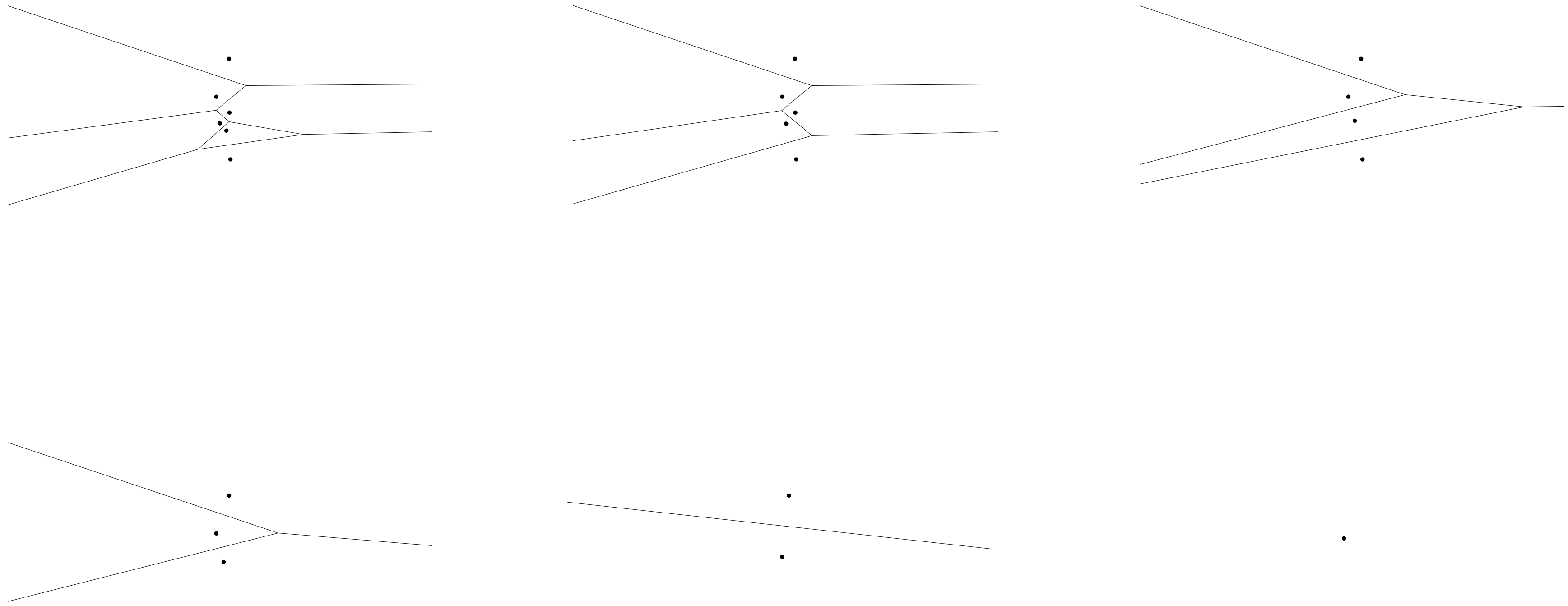
$$\mathrm{Vor}(S) = \bigcup V(p), \quad p \in S$$

- B is a nearest neighbour of A: $A \to B \Leftrightarrow \mathrm{dist}(a, b) = \min_{c \in S - a} \mathrm{dist}(a, c)$
- *Nearest neighbours search* problem: Given N points in the plane, with preprocessing, how quickly can B be found, $A \to B$ for a given B?
- Theorem:
  Nearest neighbour search can be performed with $O(\log N)$ time complexity, $O(N)$ storage complexity, and $O(N \log N)$ time complexity for preprocessing. The algorithm is optimal.

Voronoi diagram II

- The Delaunay triangulation is defined as the straight line dual graph of the Voronoi diagram.
- Theorem (H. Edelsbrunner & R. Seidel, 1986):
  The Delaunay triangulation of N points in D dimensions is the projection of $x_1, \dots, x_D$-plane of the lower convex hull of the points mapped into $(D+1)$ dimensions by $(x_1, \dots, x_D) \to (x_1, \dots, x_D, \sum_i x_i^2)$.
- Theorem (O. Devillers, S. Meiser & M. Teillaud, 1992):
  The Delaunay triangulation of N points in the plane can be dynamically maintained in $O(\log N)$ expected time to insert a point and $O(\log \log N)$ expected time to delete a point. This result holds provided that, at any time, the order of insertion on the sites remaining in S may be each order with the same probability, and when a site is deleted, it may be any site with the same probability.

FastJet III



Requirement for heavy ion jet reconstruction

- Additional requirements to k_T algorithm
  - Accomodate the probability being a jet particle (and not background) as a multiplicative weight
    - Accomodate a associated particle-only (i, j asymmetric) weight
    - Accomodate a structure dependent (both i, j dependent) weight
  - Accomodate multiple path search
- More difficult to implement efficiently

Multiplicatively weighted Voronoi diagram

- Multiplicatively weighted Voronoi regions:

$$V(p) = \{x : d_W(x,p) \leq d_W(x,q), \forall q \in S\}, \quad d_W(x,p) = |p - x|/w(p)$$

- Multiplicatively weighted Voronoi diagram MWVD(S) defined analogously to Vor(S).
- Theorem (F. Aurenhammer & H. Edelsbrunner, 1984): The MWVD(S) contains at most $O(N^2)$ edges and thus $O(N^2)$ faces and vertices.
- No published incremental deletion algorithm exists.
- Sidenote: published incremental deletion algorithm for additively weighted Voronoi diagram only works in the special case of "perturbative" weights (D. Mioc, F. Anton & C. M. Gold, 1998).

Order-k Voronoi diagram

- Generalised Voronoi polygon:

$$V(T) = \{x : |p - x| \leq |q - x|, \forall p \in T, q \in S - T\}$$

- Order-k Voronoi diagram:

$$\text{Vor}_k(S) = \bigcup V(T), \quad T \subset S, |T| = k$$

- Theorem:
  The k nearest neighbour search can be performed with $O(\log N + k)$ time complexity and $O(k^2 N \log N)$ time complexity for preprocessing.

Problems about complexity constraint

- Q1: Is it possible to construct a $O(N \log N)$ algorithm for weighted jet reconstruction, as for heavy ion events?
- A1: In the equivalent $(D+1)$ dimensional convex hull problem, a $O(N \log N)$ construction constraints the possible transform of the projective parabola to a translation in $(D+1)$-th dimension. This is only true if the weight is applied as $d \to \sqrt{w^2+d^2}$.
- Q2: What is the lowest possible complexity for a jet reconstruction with multiplicatively applied weightings?
- A2: $O(N^2)$, because of F. Aurenhammer & H. Edelsbrunner (1984).
- Q3 (open): Is it possible to perform multiplicatively weighted jet reconstruction with $O(N^2)$ preprocessing, but $O(N \log N)$ remaining operations?

$O(N^2)$ probabilistic heavy ion jet reconstruction I

- Define $n = \dfrac{\max d_{ij} \max w_i^2}{\min d_{ij}^2 \min w_i}$.
- Update order-k weighted nearest neighbourhood per step is bounded by

$$N \leq \frac{4k}{3n^2(n-1)} \left(\sqrt{3}\pi n^2 + 6n - 2\sqrt{3}\right) \left(3n^2 + 3n - 1\right)$$

$$\sim \frac{4\sqrt{3}\pi k \max d_{ij} \max w_i^2}{\min d_{ij}^2 \min w_i}$$

$$\sim O(1).$$

- Compare again with F. Aurenhammer & H. Edelsbrunner (1984): We have a algorithm that performs optimal in the geometrical worst case (while being at most $O(N/\log N)$ inefficient in best case).

$O(N^2)$ probabilistic heavy ion jet reconstruction II

- Consider a balanced tree, additionally with the external nodes being doubly linked, with an external node counter:
  - k-th item . . . . $O(k)$
  - Insertion . . . . $O(\log N)$
  - Deletion . . . . $O(\log N)$
- Leads to the following $O(N^2)$ algorithm:
  - Initial (d_{ij}) table construction with sorting, truncated at k smallest distances . . . . $O(kN^2 \log k)$
  - N steps, for each:
    - Find $\min(d_{ij}, d_{iB})$ (scan N lists k times) . . . . $O(kN)$
    - Remove protojet (tree deletion) . . . . $O(N \log k)$
    - Recalculate d_{ij} (tree insertion with overflow) . . . . $O(kN \log k)$

$O(N \log N)$ for probabilistic heavy ion jet reconstruction?

- *FastJet* circumvented the $O(N^2)$ worst-case graph complexity by imposing requirement on weight
- Originally proposed heavy ion algorithm is not compatible with this requirement
- Possible, but requires approximation $w_{ij} \approx \min(w_i, w_j)$
  - weight is symmetrized (possible for heavy ion because of monotonic p_T dependence for $p_T > 0.5\,\mathrm{GeV}/c$)
  - structure dependent probability is set to a constant

References

- G. M. Adelson-Velsky and E. M. Landis, Soviet Math. 3, 1259 (1962).
- F. Aurenhammer and H. Edelsbrunner, Pattern Recog. 17, 251 (1984).
- M. Cacciari and G. P. Salam, Laboratoire de Physique Théorique et Hautes Energies Report LPTHE-05-32, 2005 (arXiv:hep-ph/0512210).
- O. Devillers, S. Meiser, and M. Teillaud, Comp. Geom. Theor. Appl. 2, 55 (1992).
- H. Edelsbrunner and R. Seidel, Disc. Comp. Geom. 1, 25 (1986).
- S. D. Ellis and D. E. Soper, Phys. Rev. D 48, 3160 (1993).
- D. Mioc, F. Anton, and C. M. Gold, in *Proc. 14th Europ. Workshop Comp. Geom.* (Barcelona, 1998), p. 117.