Interpreting A, B, and C in a Linearized Pangenome Graph

This note explains the three colored detours in your sketch and maps each to the fields emitted by `linearizePangenome(...)`*. Add your image above/below as you prefer.*

Quick mental model

- **Spine (reference ribbon):** the pink path you chose (e.g., GRCh38). In the linearized view it sits at **y = 0**, and **x** is proportional to base pairs.
- **Detour / Bubble / Loop:** any alternative path that leaves the spine at a **left anchor** `L` and rejoins at a **right anchor** `R`. Each detour is returned as one object in `loops[]`.

For every detour the function computes:

- `spanStart` = `bpEnd(L)` and `spanEnd` = `bpStart(R)` → the **reference span being replaced**.
- `refLenBp = spanEnd − spanStart` → length of that reference span.
- `altLenBp` → sum of node lengths along the colored detour.
- `apex` / `lane` → screen-space vertical placement to prevent overlaps (no genomic meaning).
- `insertionLike / deletionLike / pill` → convenience flags for styling.

A — Insertion-like bubble (same orientation)

- **Description:** A simple detour between anchors `L` and `R` where the alt path is **longer** than the replaced reference span (in your sketch, teal under the spine).
- **Fields in `loops[]`:**
  - `leftId = L`, `rightId = R`
  - `spanStart = bpEnd(L)`, `spanEnd = bpStart(R)`
  - `refLenBp = spanEnd − spanStart`
  - `altLenBp = Σ lengthBp along the teal detour`
  - `insertionLike: true`, `deletionLike: false`, `pill: false`
  - With the default convention, the loop is placed **above** the spine (`apex > 0`).

B — Inversion-like detour (reversed orientation)

- **Description:** An alternative path between `L` and `R` whose **arrow direction is opposite** the local reference direction—i.e., an **inversion** across that span (blue in your sketch).
- **Fields in `loops[]`:**
  - `leftId = L`, `rightId = R`
  - `spanStart`, `spanEnd`, `refLenBp` as above
  - `altLenBp = Σ lengthBp along the blue detour`

- *(Optional)* `inversionLike: true` if you add an orientation check in your pipeline

- **Styling tip:** Keep the above/below rule based on `altLenBp` vs `refLenBp`, and add a visual cue for inversion (e.g., dashed loop, "twist" glyph at the apex, or reversed arrowheads on the detour).

C — Deletion-like bubble (same orientation)

- **Description:** A simple detour between `L` and `R` where the alt path is **shorter** than the replaced reference span (green above the left branch in your sketch).

- **Fields in `loops[]`:**

  - `leftId = L`, `rightId = R`
  - `spanStart`, `spanEnd`, `refLenBp` as above
  - `altLenBp = Σ lengthBp along the green detour`
  - `insertionLike: false`, `deletionLike: true`, `pill: false`
  - With the default convention, the loop is placed **below** the spine (`apex < 0`).

Glossary (field ↔ concept)

Field	Meaning
`spineSegments[]`	Straight segments of the reference (pink), laid out length-true at `y = 0`.
`leftId`, `rightId`	The two spine nodes that anchor a detour.
`spanStart`, `spanEnd`	Spine bp coordinates for where the detour leaves and rejoins (`bpEnd(L)`, `bpStart(R)`).
`refLenBp`	Length of the replaced reference span (`spanEnd − spanStart`).
`altLenBp`	Total base-pair length along the detour path.
`pill`	`true` when `refLenBp == 0` (anchored insertion at one position); drawn as a small vertical "pill."
`insertionLike`	`altLenBp > refLenBp`. Conventionally drawn **above** the spine.
`deletionLike`	`altLenBp < refLenBp`. Conventionally drawn **below** the spine.
`apex`, `lane`	Screen-space vertical placement used to avoid overlaps.
`bezier` / `points`	Geometry for drawing the loop (cubic Bézier control points or a sampled polyline).

Optional: detecting inversions

If your node/edge IDs carry orientation (e.g., `12345+` / `12345−`) or your edges encode direction, you can set an `inversionLike` flag by checking whether the detour's net direction between `L` and `R` opposes the spine's direction over the same span. Use that flag to style cases like **B** distinctly without changing their above/below placement.

Conventions used by the function (and in this doc):

- Above = **insertion-like** (`altLenBp > refLenBp`), below = **deletion-like**.
- Horizontal width = **reference span** being replaced; vertical offset = **layout only**.
- Pills appear when the two anchors coincide (`refLenBp == 0`).