Preserving topology while sampling

Trials and tribulations

Preserving topology while sampling

Trials and tribulations

Andrew Rechnitzer   Nick Beaton   Nathan Clisby

February 2022 — Richard Brak

The question(s)

- How does topology influence geometry?
- What does a trefoil look like?
- Which trefoil?

The question(s)

- How does topology influence geometry?
- What does a trefoil look like?
- Which trefoil?

So *"just"*

- Define a probability measure on the set of closed curves in $\mathbb{R}^3$
- Use that to study a typical trefoil

The question(s)

- How does topology influence geometry?
- What does a trefoil look like?
- Which trefoil?

So *"just"*

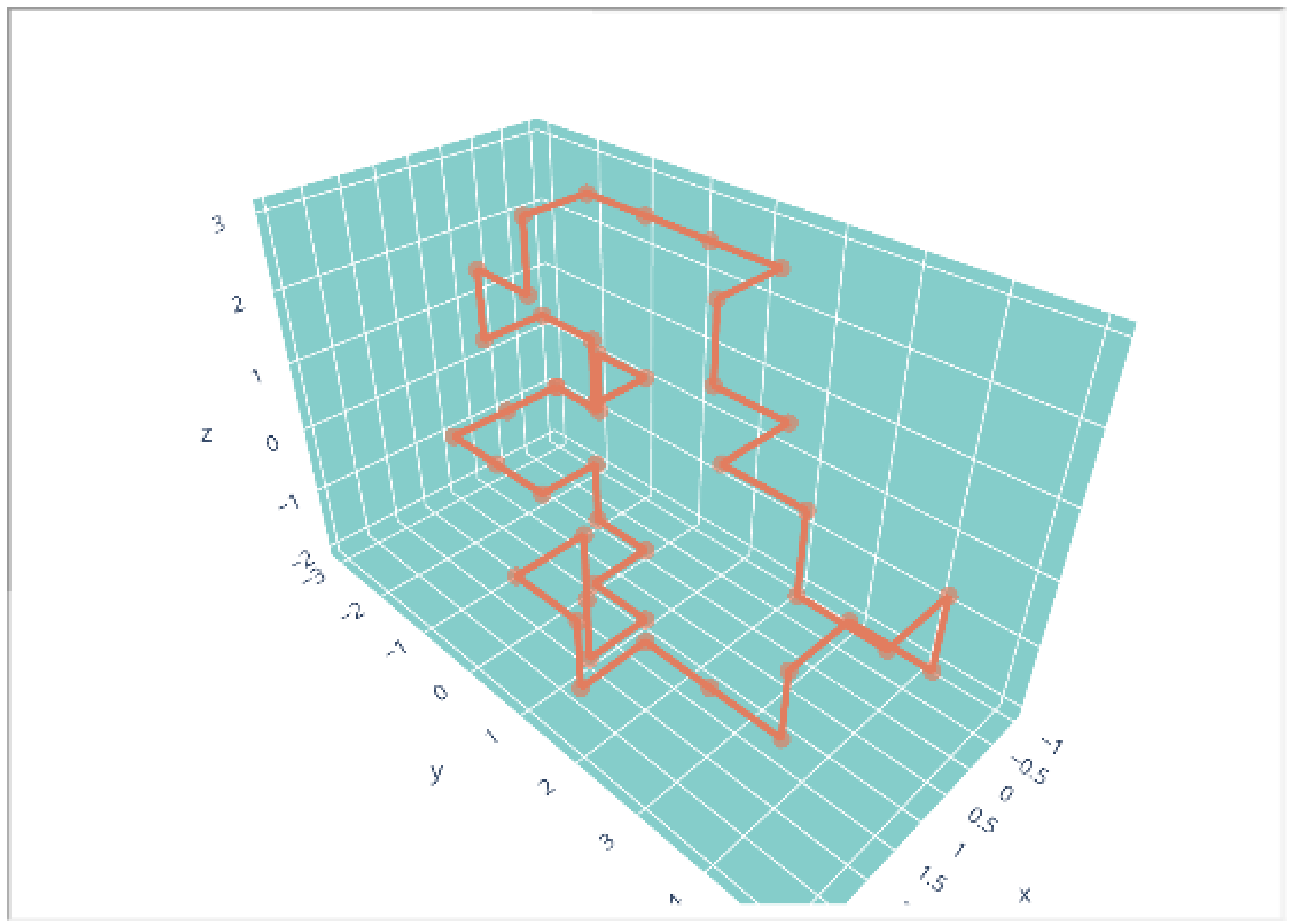
- Define a probability measure on the set of closed curves in $\mathbb{R}^3$
- Use that to study a typical trefoil

How hard could it be?

My favourite two measures

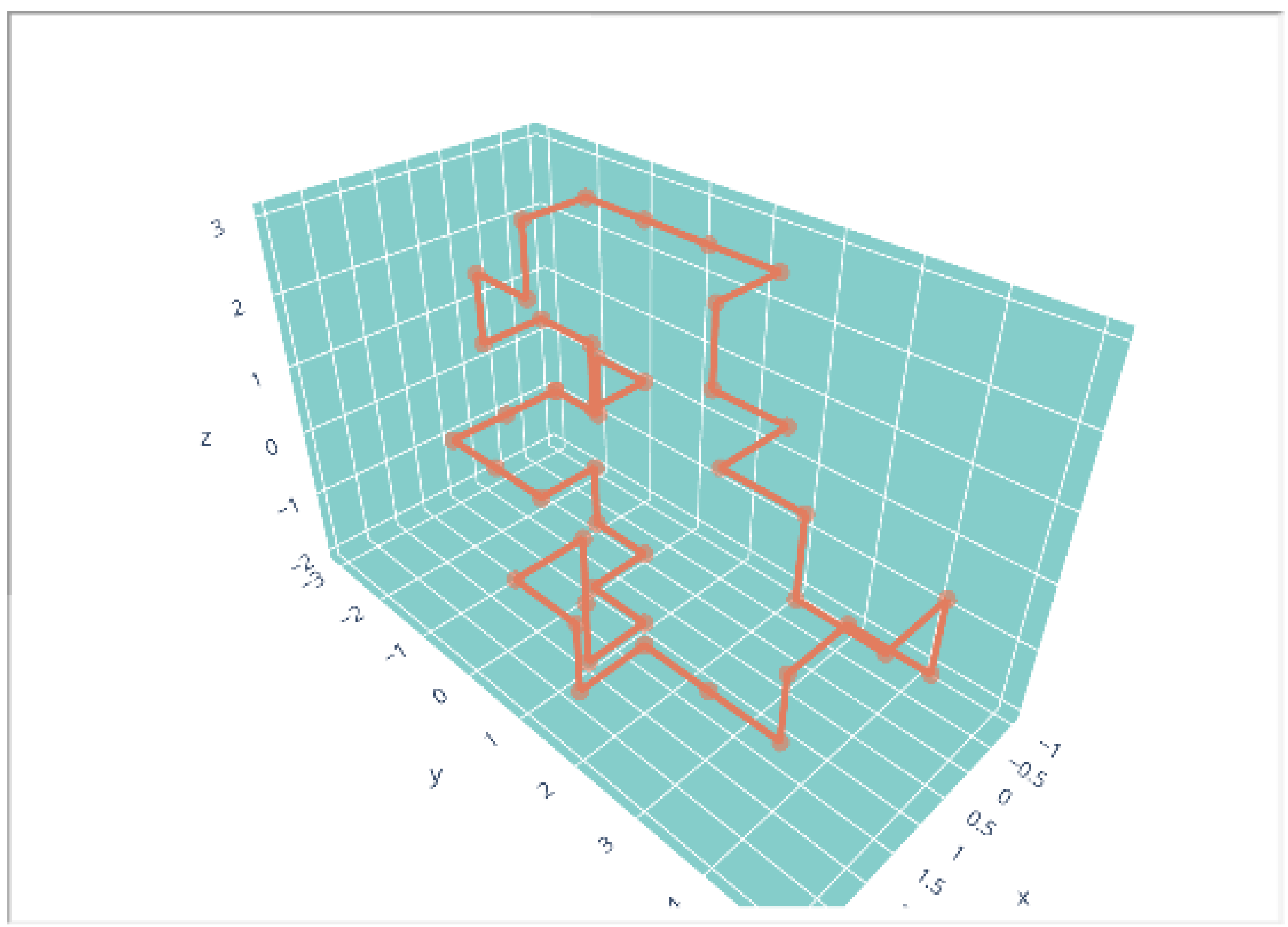
My favourite two measures

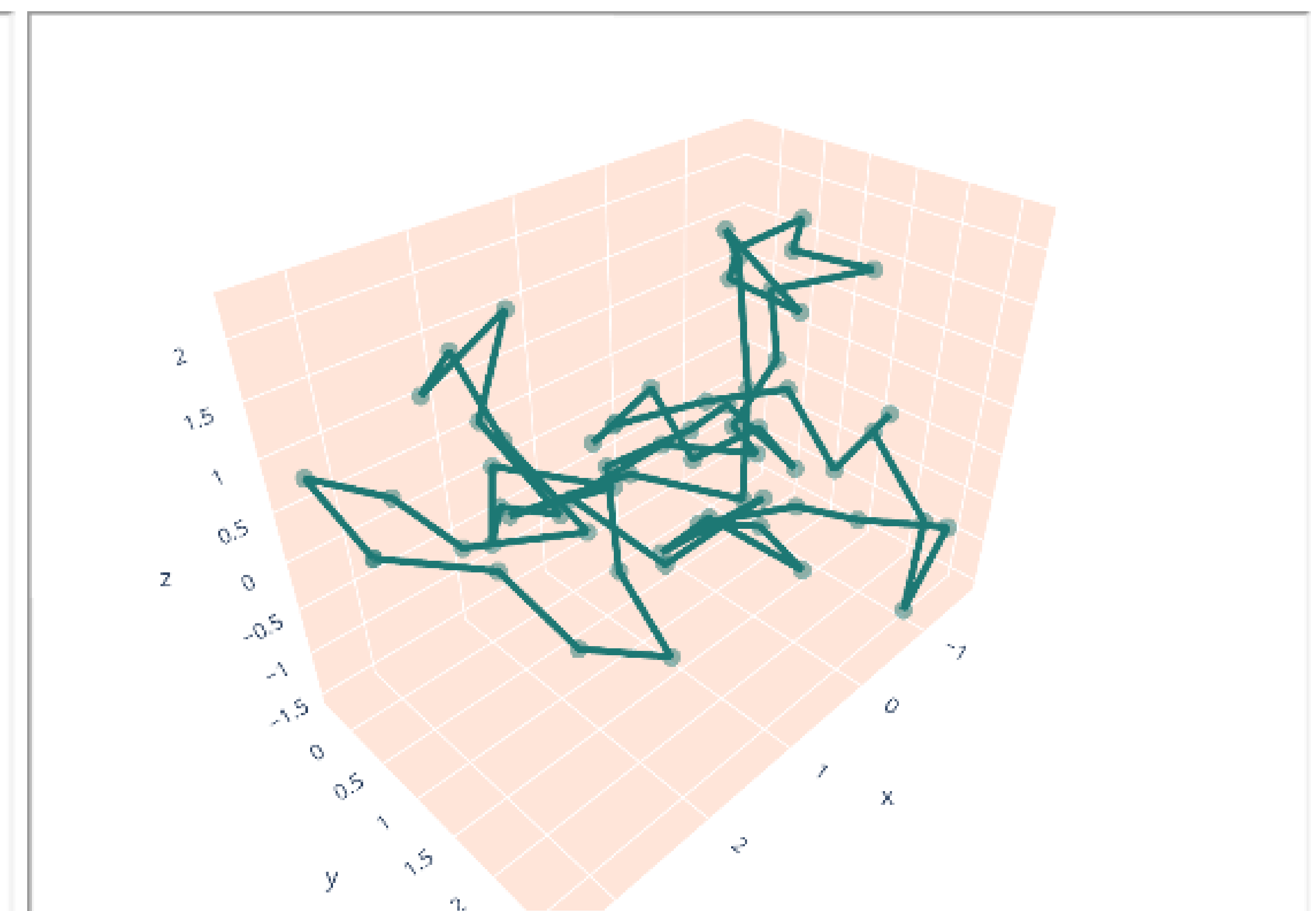
- Self-avoiding polygons (SAP) in $\mathbb{Z}^3$
  - embedding of simple loop into lattice
  - each embedding of length n equally likely



My favourite two measures

- Self-avoiding polygons (SAP) in $\mathbb{Z}^3$
  - embedding of simple loop into lattice
  - each embedding of length n equally likely
- Equilateral random polygons (ERP) in $\mathbb{R}^3$
  - each edge has unit length
  - edge direction chosen uniformly on S^2, conditioned to close





Analytic results are very hard

- Work by Whittington, Sumners, Millett, Soteros, van Rensburg, Orlandini, Deguchi, Cantarella, Micheletti, Grosberg, …
- Please see this excellent review with a much more complete list

Analytic results are very hard

- Work by Whittington, Sumners, Millett, Soteros, van Rensburg, Orlandini, Deguchi, Cantarella, Micheletti, Grosberg, …
- Please see this excellent review with a much more complete list

Resort to random sampling instead

Analytic results are very hard

- Work by Whittington, Sumners, Millett, Soteros, van Rensburg, Orlandini, Deguchi, Cantarella, Micheletti, Grosberg, …
- Please see this excellent review with a much more complete list

Resort to random sampling instead

- Sample a superset and then sieve out the ones you want, or

Analytic results are very hard

- Work by Whittington, Sumners, Millett, Soteros, van Rensburg, Orlandini, Deguchi, Cantarella, Micheletti, Grosberg, …
- Please see this excellent review with a much more complete list

Resort to random sampling instead

- Sample a superset and then sieve out the ones you want, or
- Sample only curves of the given fixed topology

Sample superset then sieve #1

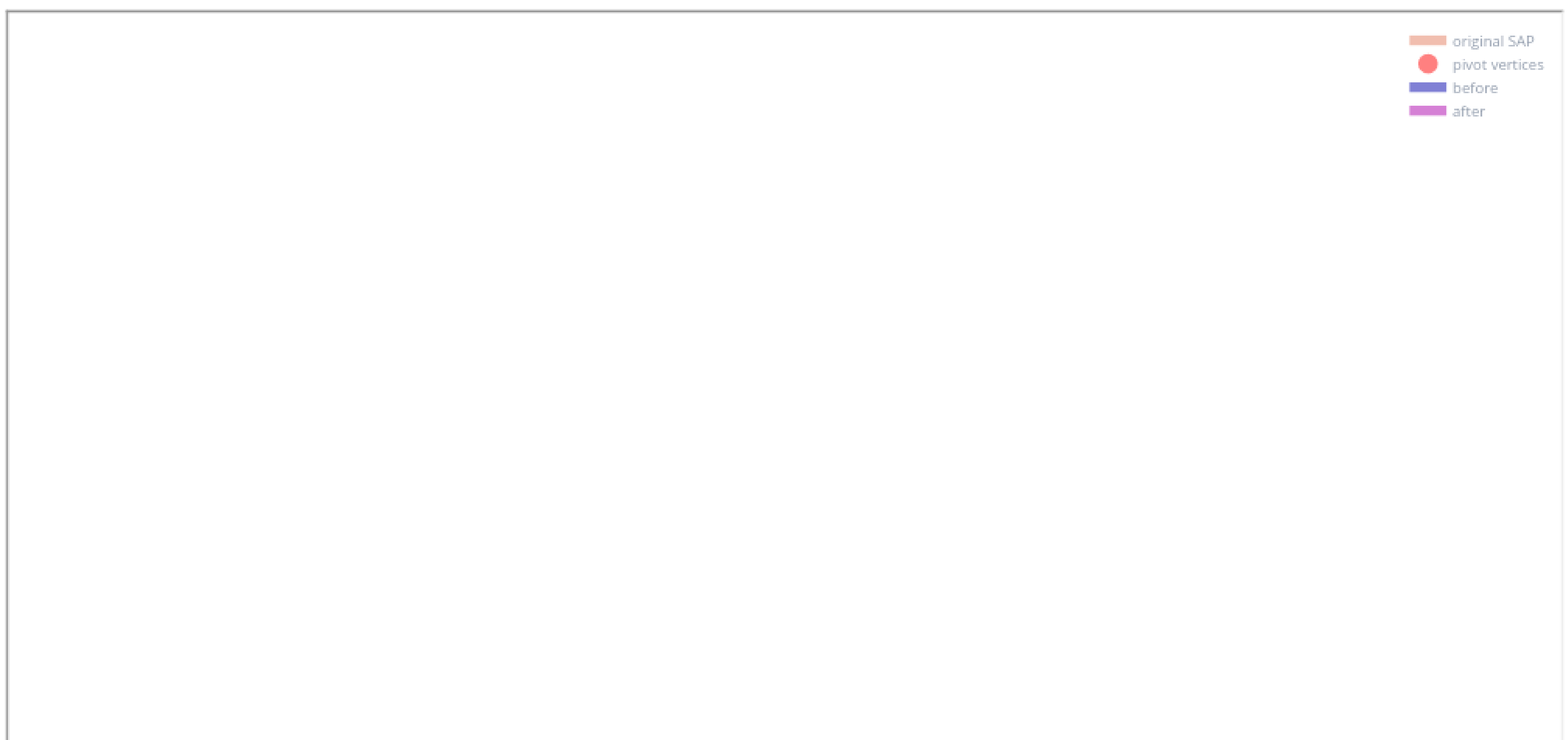
- Exact random sampling of ERP — Cantarella et al (2015)
- Time $O(n^{5/2})$ to produce an completely independent ERP

Sample superset then sieve #1

- Exact random sampling of ERP — Cantarella et al (2015)
- Time $O(n^{5/2})$ to produce an completely independent ERP

Sample superset then sieve #2

- Pivot algorithm on SAP of fixed length — Lai (1969), Madras & Sokal (1988), Madras et al (1990)
- Clisby (2010) implementation — $O(\log n)$ to sample statistically "*independent*" walk



Topological testing

- Unknot identification is hard — Hass et al (1999)
- Knot invariants are slow to compute

Topological testing

- Unknot identification is hard — Hass et al (1999)
- Knot invariants are slow to compute
- Polygons of given topology become exponentially rare as length grows — Sumners & Whittington (1988), Pippenger (1989)

Topological testing

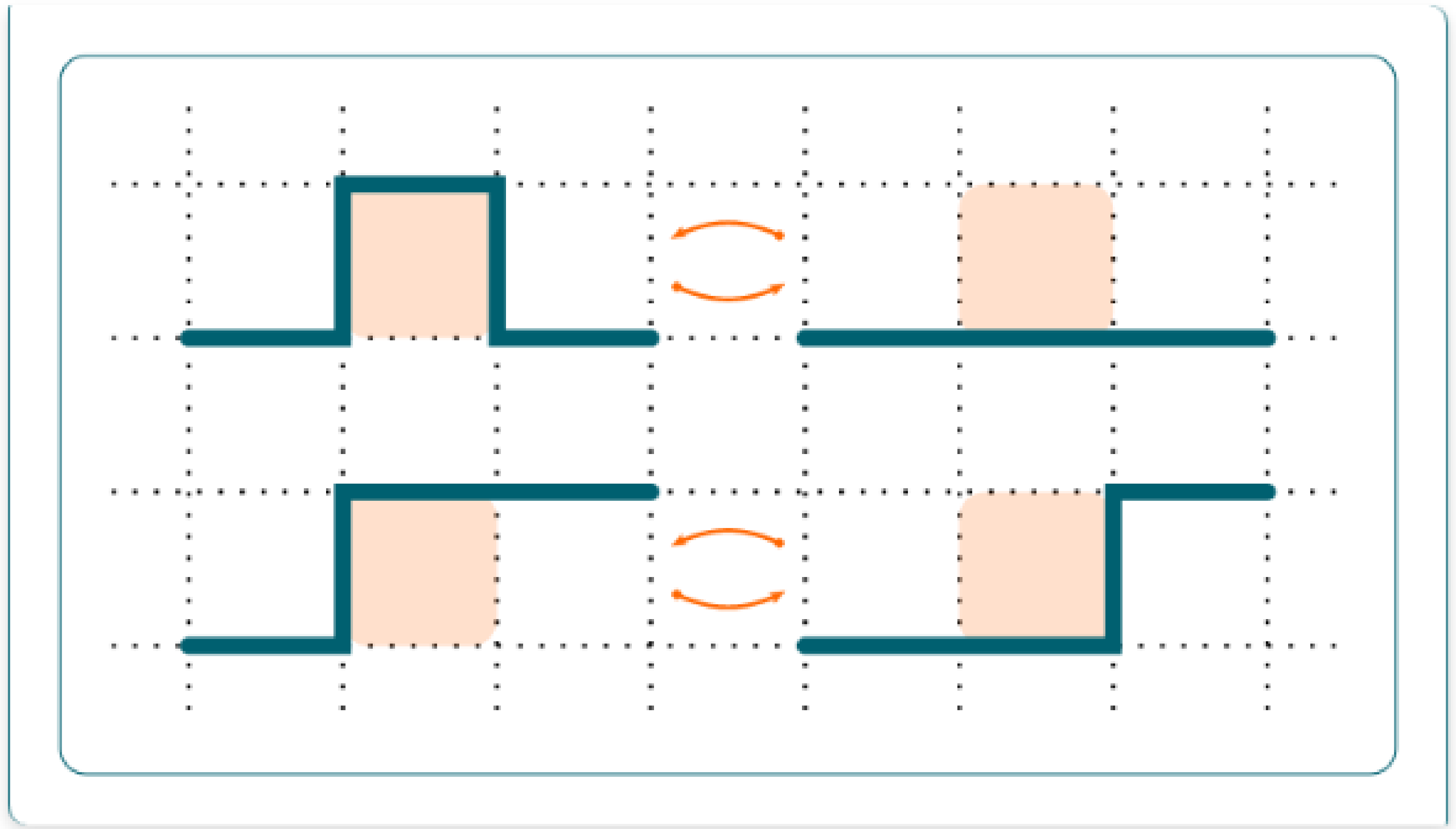
- Unknot identification is hard — Hass et al (1999)
- Knot invariants are slow to compute
- Polygons of given topology become exponentially rare as length grows — Sumners & Whittington (1988), Pippenger (1989)
- Identification is the bottleneck when sampling long polygons
    - long polygon $\Longrightarrow$ many crossings $\Longrightarrow$ hard to ID

Topological testing

- Unknot identification is hard — Hass et al (1999)
- Knot invariants are slow to compute
- Polygons of given topology become exponentially rare as length grows — Sumners & Whittington (1988), Pippenger (1989)
- Identification is the bottleneck when sampling long polygons
  - long polygon $\Longrightarrow$ many crossings $\Longrightarrow$ hard to ID
- Aside — how can we measure the trefoilness of a larger knot?

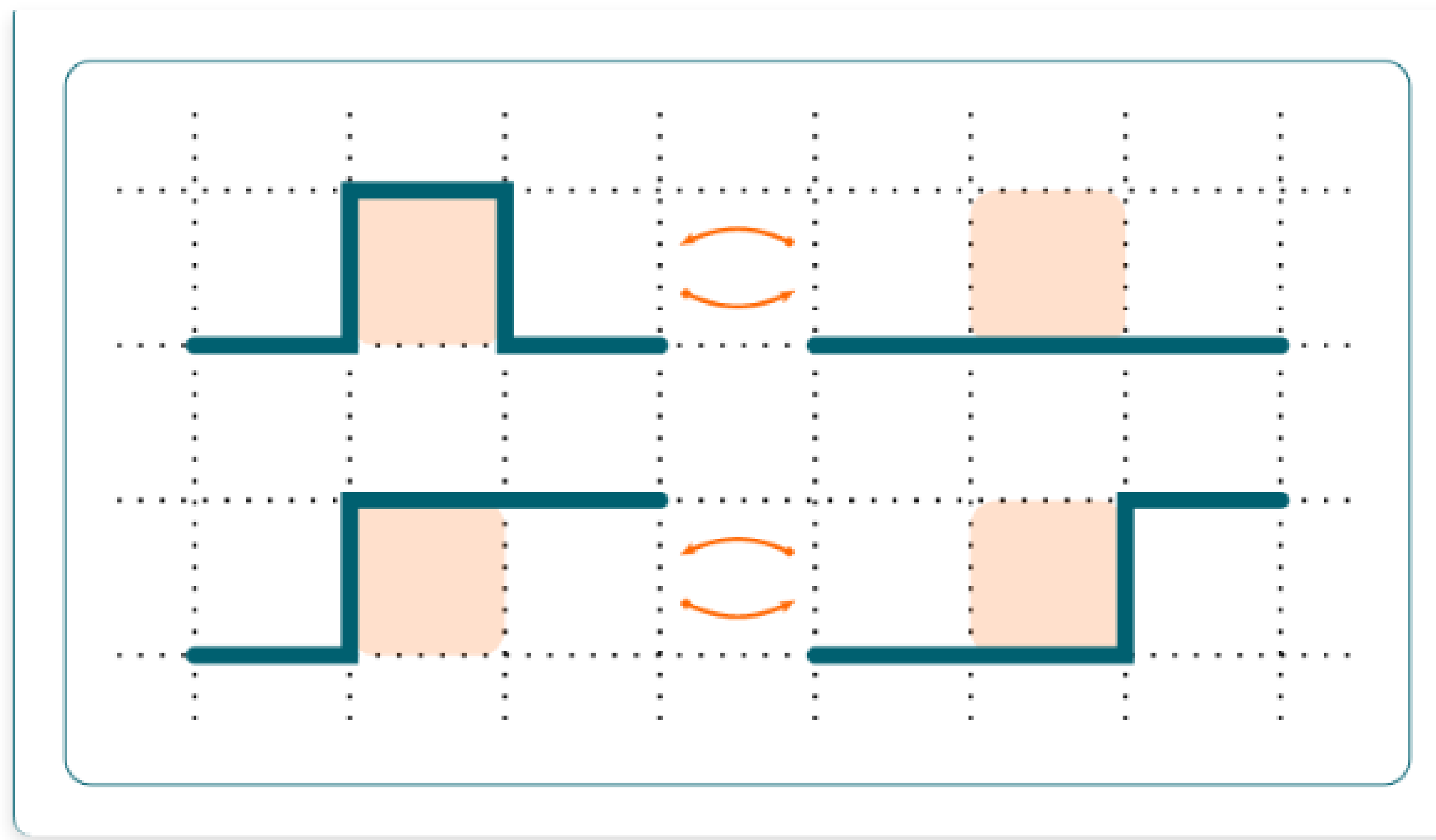
Sample only fixed topology #1

- Markov chain on SAPs of fixed topology — B.F. A.C.F. (1981, 1983)
- No topological testing needed — strand passage not possible
- Ergodic on knot type van Rensburg & Whittington (1991), van Rensburg & R (2011)



Sample only fixed topology #1

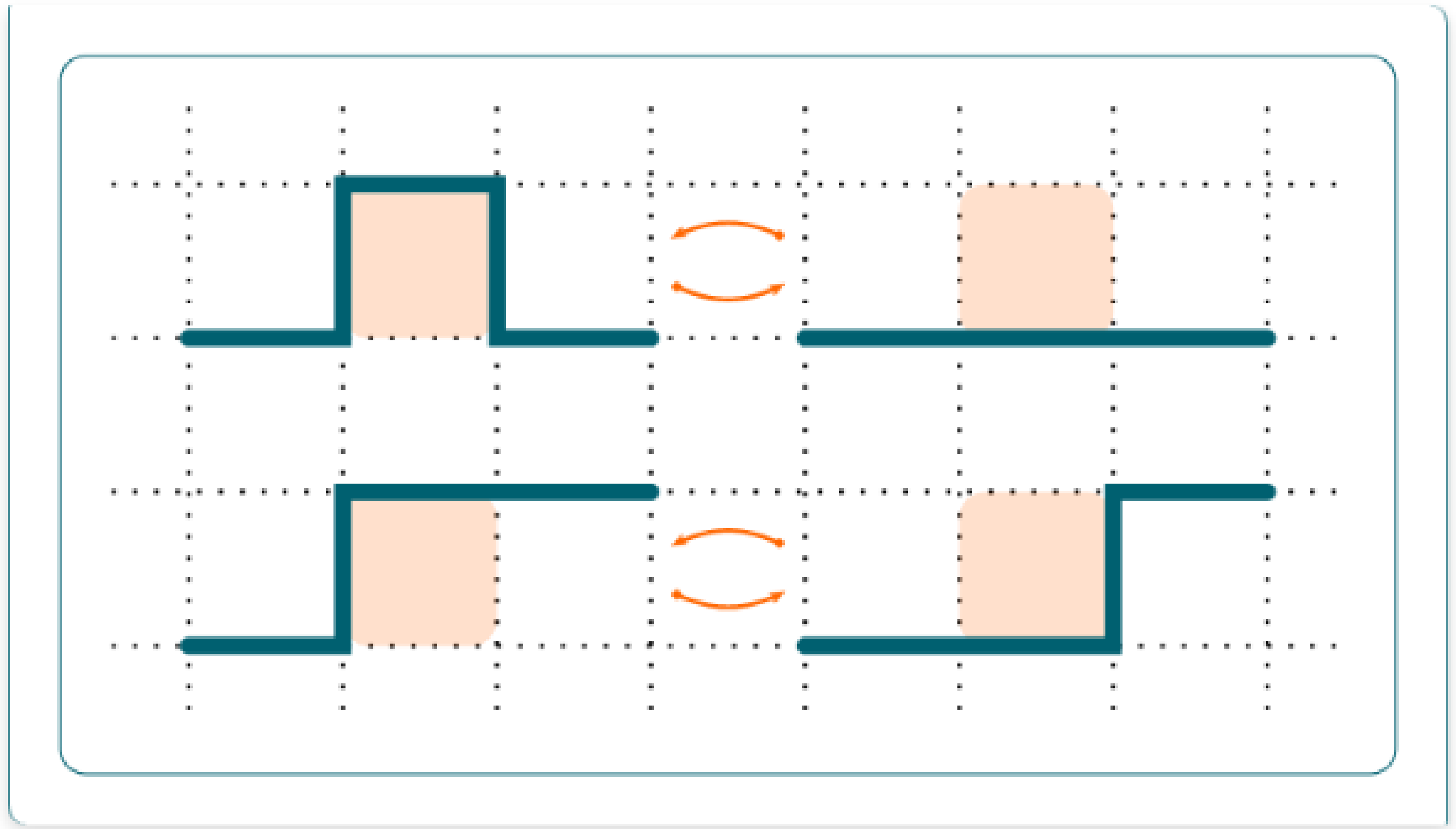
- Markov chain on SAPs of fixed topology — B.F. A.C.F. (1981, 1983)
- No topological testing needed — strand passage not possible
- Ergodic on knot type van Rensburg & Whittington (1991), van Rensburg & R (2011)



- Start with small conformation — deform with local moves
- Tune so that grow/shrink moves equally likely to succeed

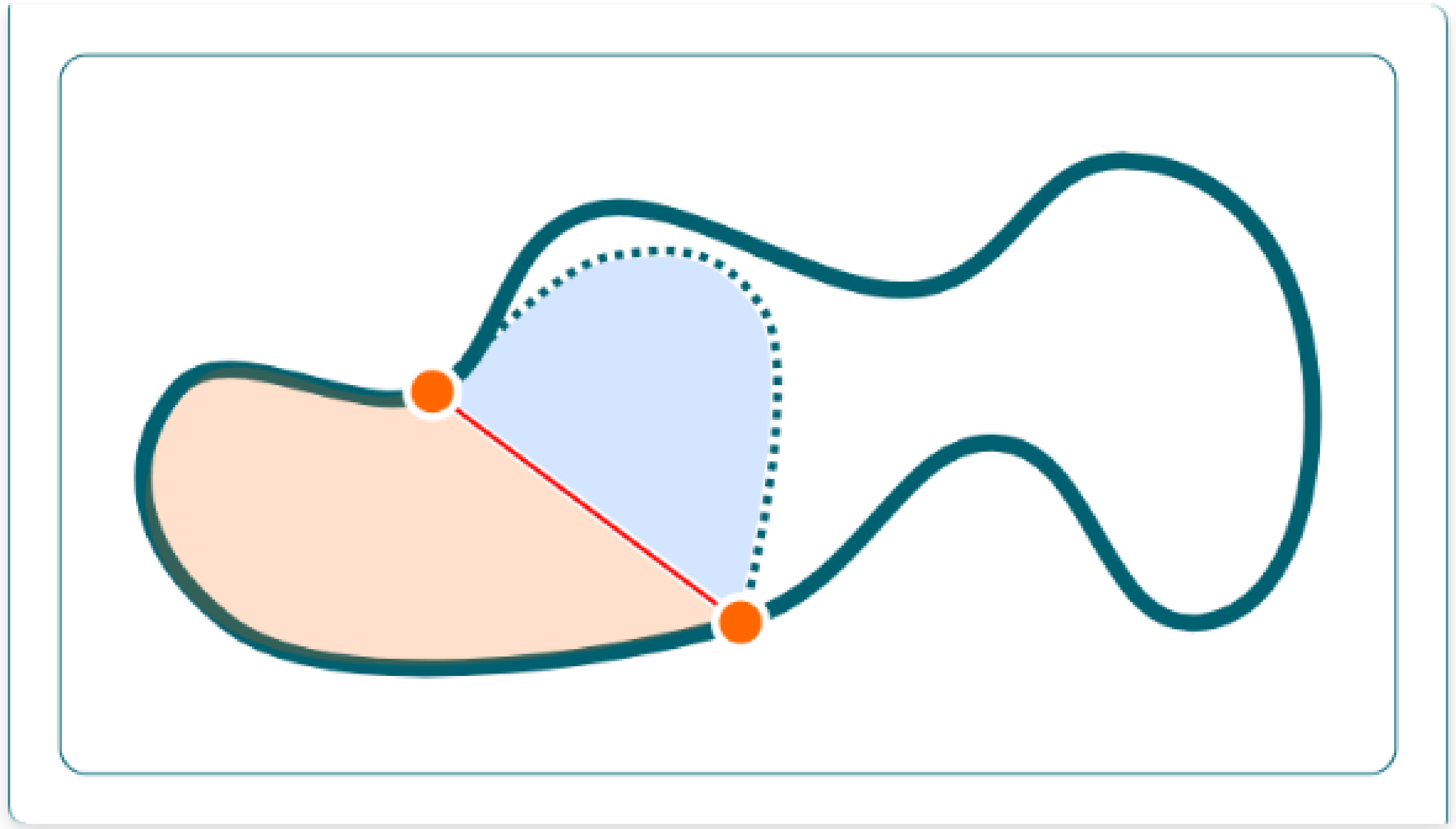
Sample only fixed topology #1

- Markov chain on SAPs of fixed topology — B.F. A.C.F. (1981, 1983)
- No topological testing needed — strand passage not possible
- Ergodic on knot type van Rensburg & Whittington (1991), van Rensburg & R (2011)



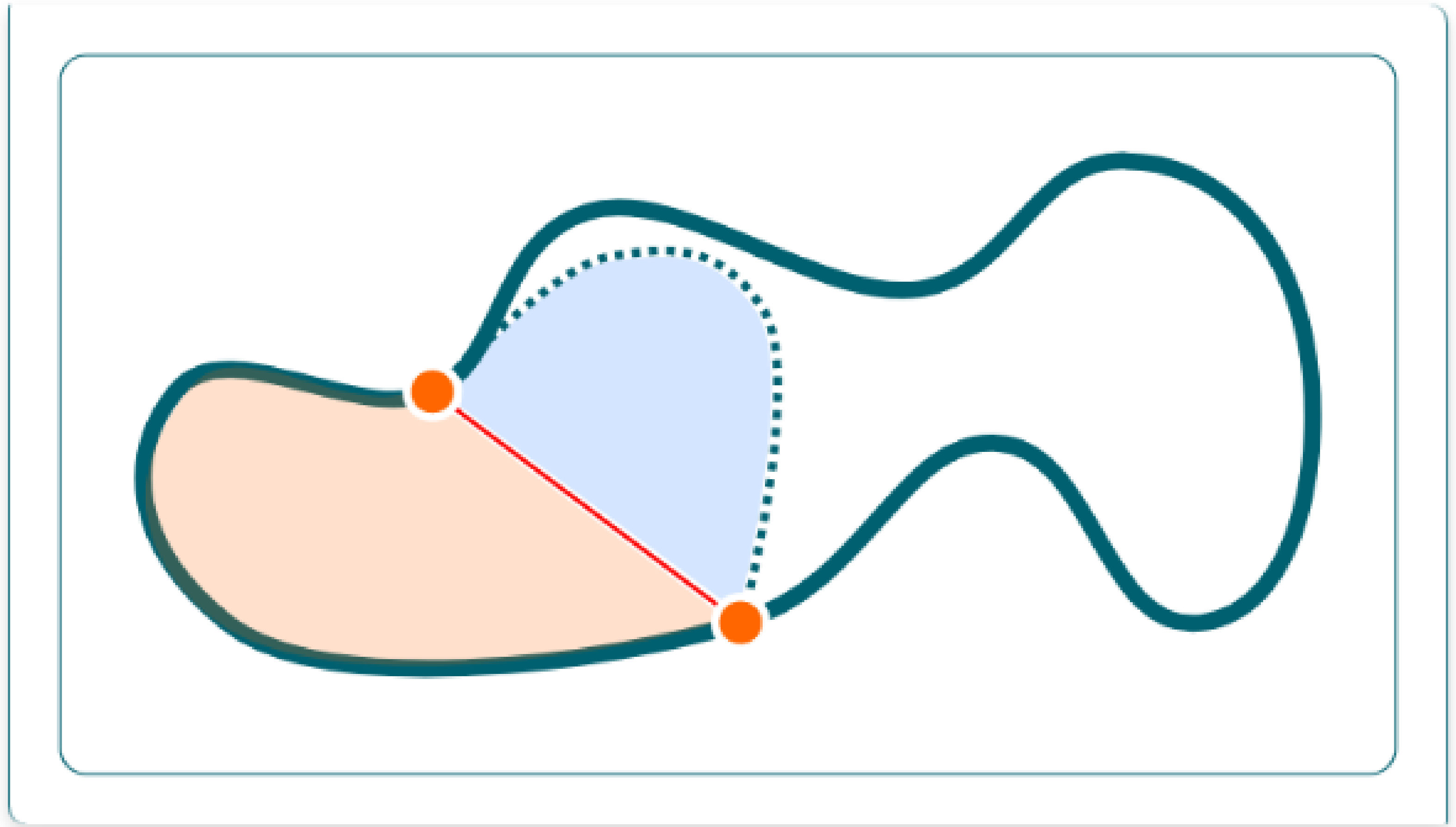
- Start with small conformation — deform with local moves
- Tune so that grow/shrink moves equally likely to succeed
- Random walk on polygon length — long time to sample *"independent"* long polygons

Fixed topology #2 — restricted pivots



- Pivot with excluded area algorithm Zhao & Ferrari (2012)
- Attempt pivot segment $\Phi \mapsto \Phi'$
- Pivot fails if edge crosses surface bordered by $\Phi \cup \Phi'$

Fixed topology #2 — restricted pivots



- Pivot with excluded area algorithm Zhao & Ferrari (2012)
- Attempt pivot segment $\Phi \mapsto \Phi'$
- Pivot fails if edge crosses surface bordered by $\Phi \cup \Phi'$
- Computationally intensive — only allowed short segment $|\Phi| \leq 5$
- Probably *"okay"* for moderate size polygons — but not ergodic Madras & Sokal (1987)

So what can we do to speed things up?

Revisit pivots again — reduce computation

Revisit pivots again — reduce computation

- Change how we think of pivots to simplify topology checking
  - need not *"literally"* pivot the segment about the axis
  - a pivot will be a continuous deformation

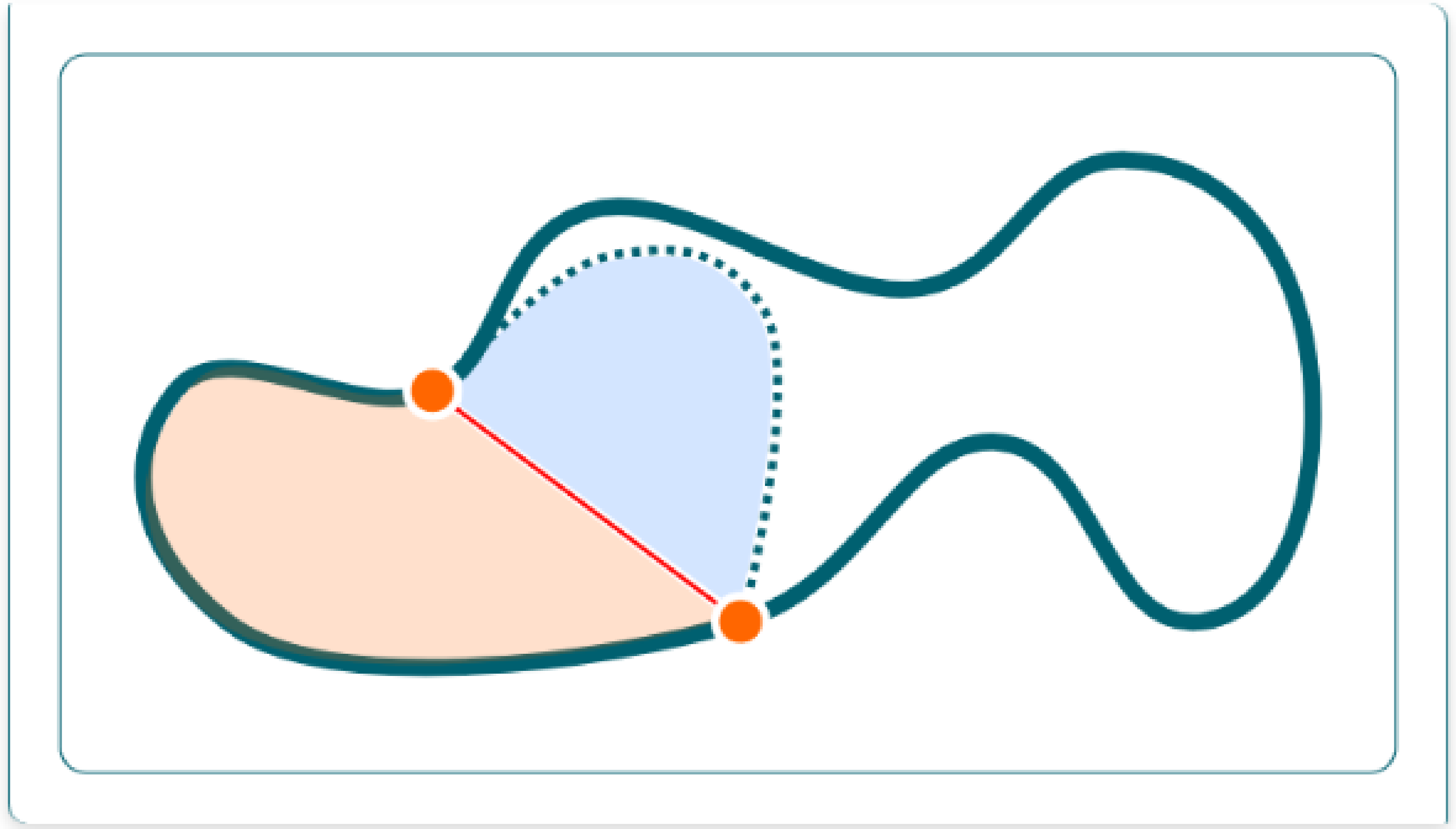
Revisit pivots again — reduce computation

- Change how we think of pivots to simplify topology checking
  - need not *"literally"* pivot the segment about the axis
  - a pivot will be a continuous deformation
- Use Clisby method for $O(\log n)$ pivots
  - Store polygon and symmetries in binary tree
  - Lazy evaluation of observables — don't write down the polygon

Revisit pivots again — reduce computation

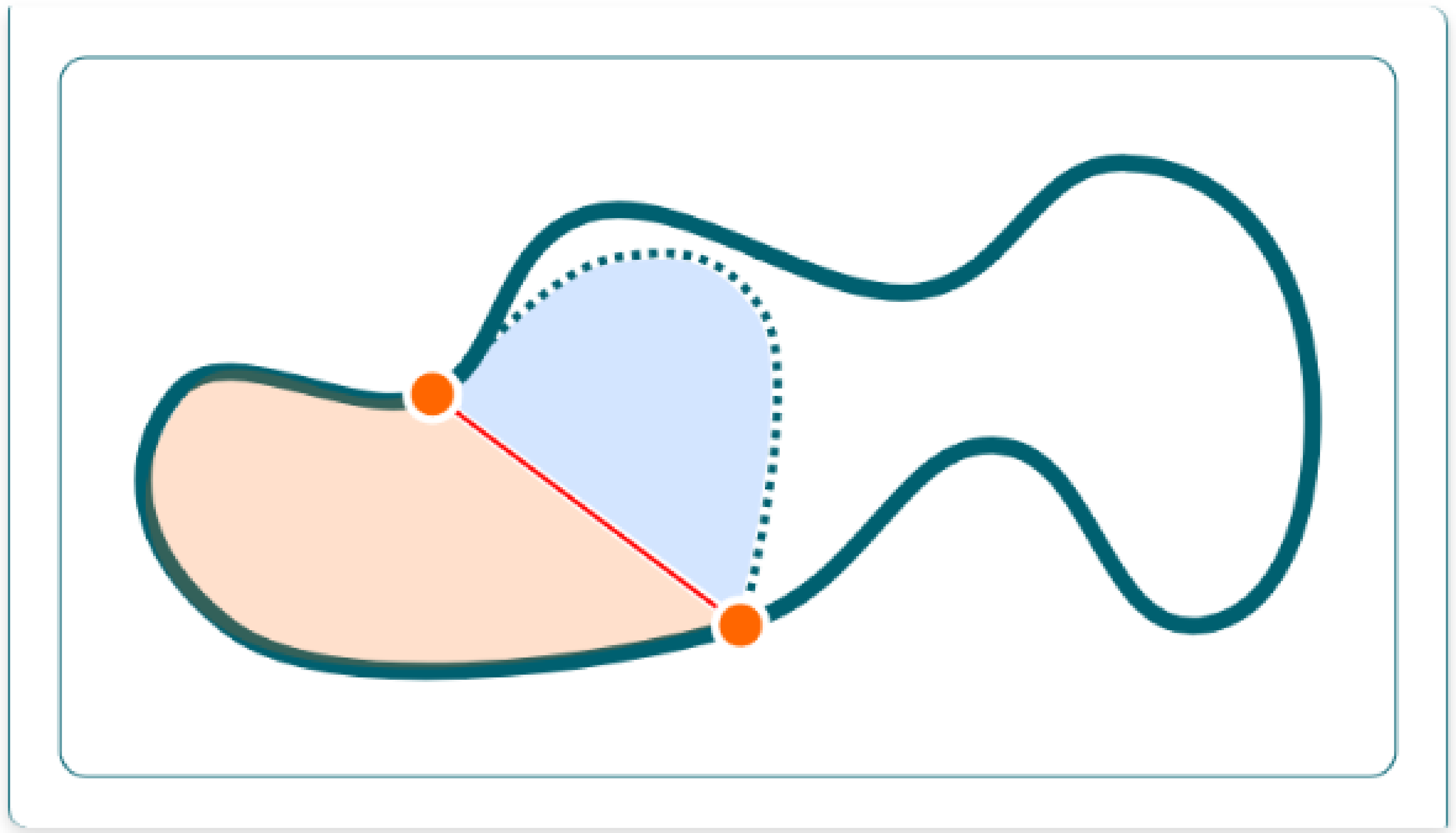
- Change how we think of pivots to simplify topology checking
  - need not *"literally"* pivot the segment about the axis
  - a pivot will be a continuous deformation
- Use Clisby method for $O(\log n)$ pivots
  - Store polygon and symmetries in binary tree
  - Lazy evaluation of observables – don't write down the polygon
- Aside – this is actually not so far from Cantarellean encoding of polygons via triangulations

Inner pivot



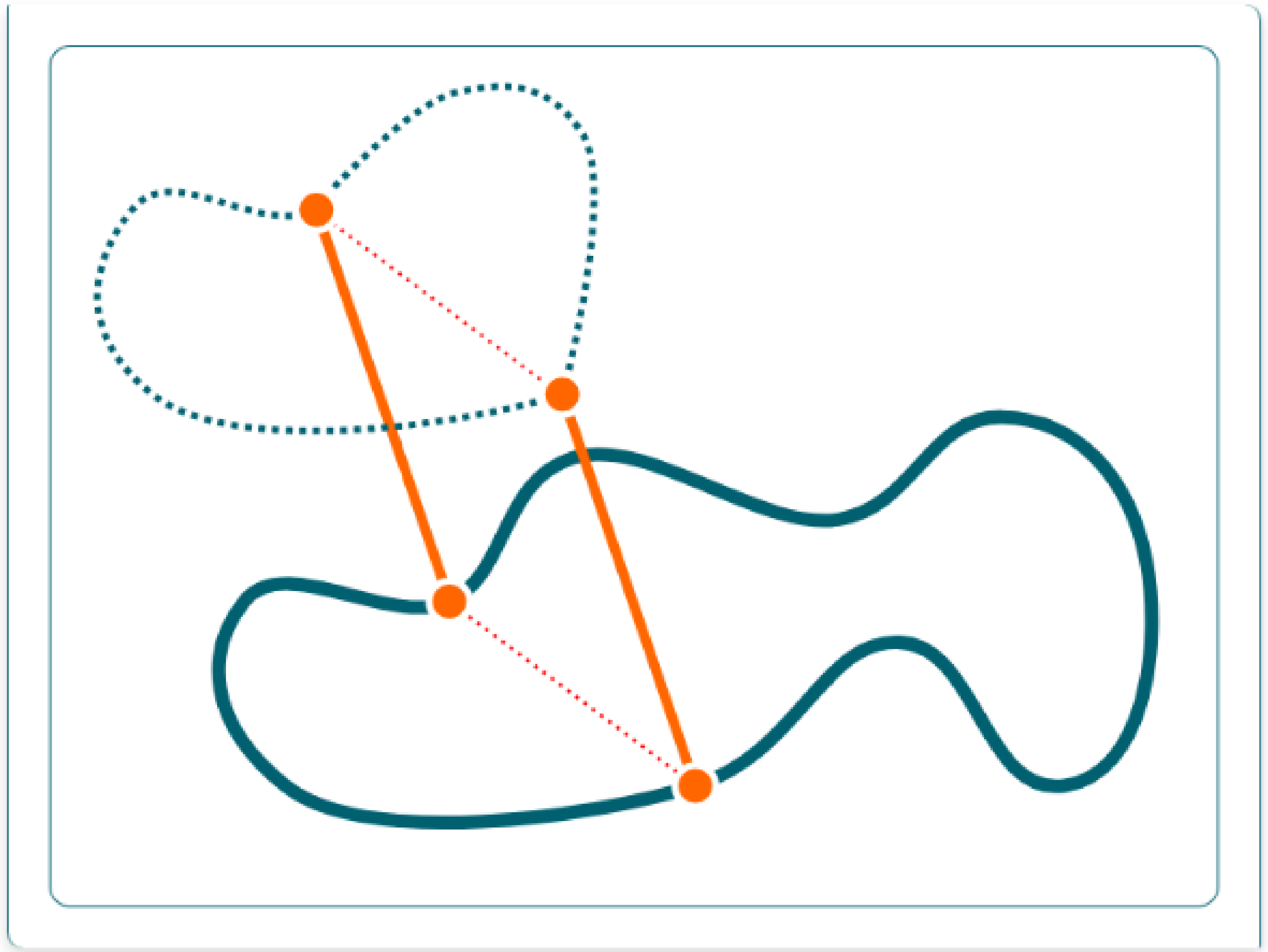
- Pick pivot segment and rotation angle

Inner pivot



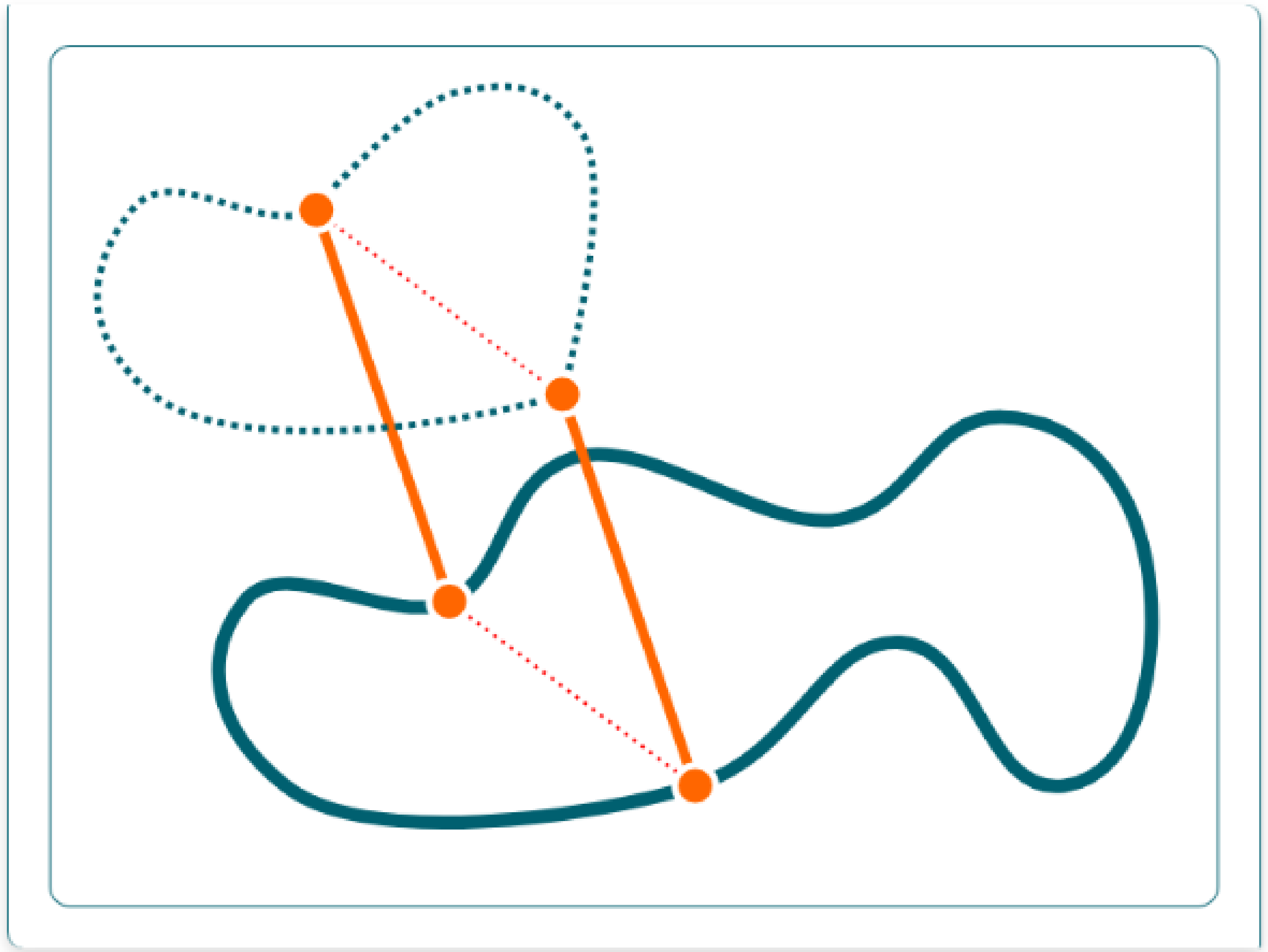
- Pick pivot segment and rotation angle
- Topology checking
  - Each pivot edge maps out a twisted quadrilateral
  - Check intersection of fixed edges with triangulation of those quadrilaterals
  - Use ray-tracing methods — eg Möller-Trumbore (1997)

Outer pivot



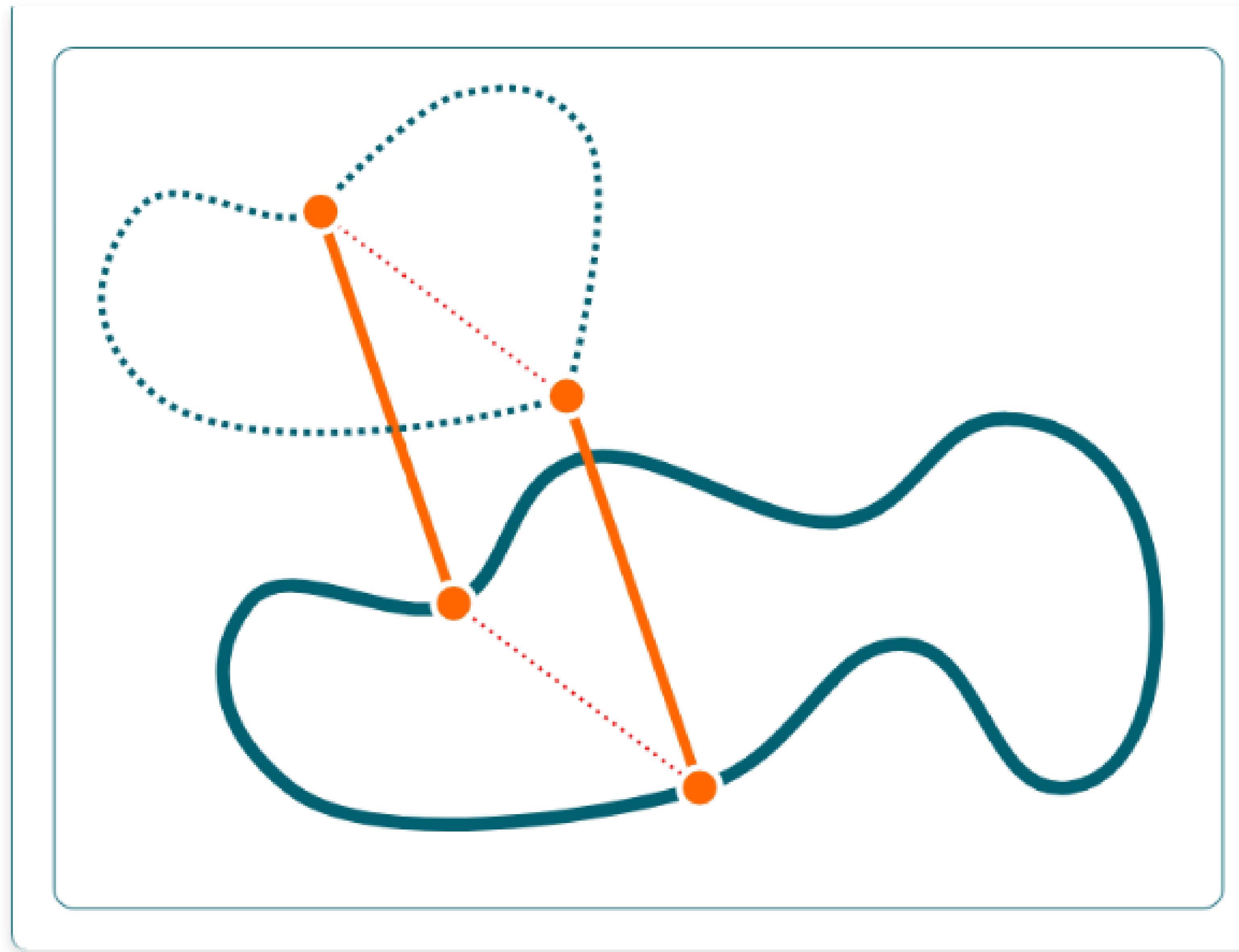
- Pick the pivot segment and an orthogonal drag direction

Outer pivot



- Pick the pivot segment and an orthogonal drag direction
  - drag the segment to infinity
  - pivot the segment at infinity
  - drag the segment back from infinity

Outer pivot



- Pick the pivot segment and an orthogonal drag direction
  - drag the segment to infinity
  - pivot the segment at infinity
  - drag the segment back from infinity
- Topology checking
  - drag to/from infinity $\mapsto$ segment overlap in projection
  - pivot at infinity $\mapsto$ check intersection with drag lines

Simple implementation of inner and outer pivots

- Computation time is $O(n^2)$ or $O(n \log n)$:
  - pick pivot vertices: $O(1)$ on $\mathbb{R}^3$
  - inner pivot: naive $O(n^2)$, but maybe as fast as $O(n \log n)$?
  - drag to infinity: naive $O(n^2)$, or Shamos-Hoey (1976) $O(n \log n)$
  - pivot at infinity: naive $O(n)$
  - write down new polygon $O(n)$

Simple implementation of inner and outer pivots

- Computation time is $O(n^2)$ or $O(n \log n)$:
  - pick pivot vertices: $O(1)$ on $\mathbb{R}^3$
  - inner pivot: naive $O(n^2)$, but maybe as fast as $O(n \log n)$?
  - drag to infinity: naive $O(n^2)$, or Shamos-Hoey (1976) $O(n \log n)$
  - pivot at infinity: naive $O(n)$
  - write down new polygon $O(n)$
- When pivot fails it fails quickly

Simple implementation of inner and outer pivots

- Computation time is $O(n^2)$ or $O(n \log n)$:
  - pick pivot vertices: $O(1)$ on $\mathbb{R}^3$
  - inner pivot: naive $O(n^2)$, but maybe as fast as $O(n \log n)$?
  - drag to infinity: naive $O(n^2)$, or Shamos-Hoey (1976) $O(n \log n)$
  - pivot at infinity: naive $O(n)$
  - write down new polygon $O(n)$
- When pivot fails it fails quickly
- When it succeeds makes a big change to the conformation

Simple implementation of inner and outer pivots

- Computation time is $O(n^2)$ or $O(n \log n)$:
  - pick pivot vertices: $O(1)$ on $\mathbb{R}^3$
  - inner pivot: naive $O(n^2)$, but maybe as fast as $O(n \log n)$?
  - drag to infinity: naive $O(n^2)$, or Shamos-Hoey (1976) $O(n \log n)$
  - pivot at infinity: naive $O(n)$
  - write down new polygon $O(n)$
- When pivot fails it fails quickly
- When it succeeds makes a big change to the conformation
- Autocorrelation time?

Clisbification by analogy

Clisbification by analogy

- Consider the product $q = x^a y^b z^c$
  - Numbers $x, y, z \in \mathbb{R}$ changed rarely
  - Numbers $a, b, c \in \mathbb{N}$ changed often
  - How should you compute the product?

Clisbification by analogy

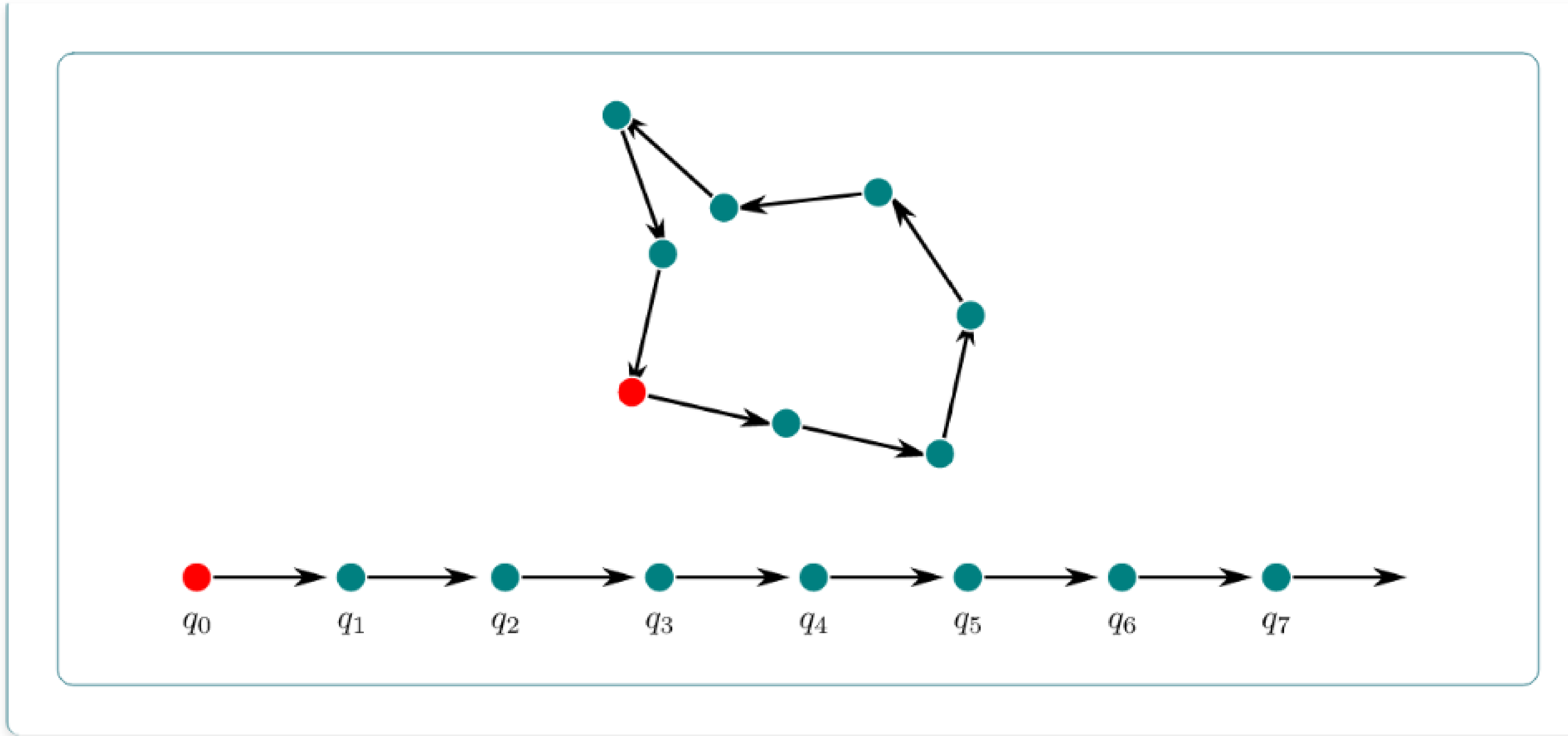
- Consider the product $q = x^a y^b z^c$
  - Numbers $x, y, z \in \mathbb{R}$ changed rarely
  - Numbers $a, b, c \in \mathbb{N}$ changed often
  - How should you compute the product?
- Standard sneaky logarithmic trick
  - When y changes, pre-compute $y^2, y^4, y^8, y^{16}, \ldots$
  - Then find y^b as product of pre-computed powers

Clisbification by analogy

- Consider the product $q = x^a y^b z^c$
  - Numbers $x, y, z \in \mathbb{R}$ changed rarely
  - Numbers $a, b, c \in \mathbb{N}$ changed often
  - How should you compute the product?
- Standard sneaky logarithmic trick
  - When y changes, pre-compute $y^2, y^4, y^8, y^{16}, \ldots$
  - Then find y^b as product of pre-computed powers
- Careful precomputation and lazy evaluation

Clisbification

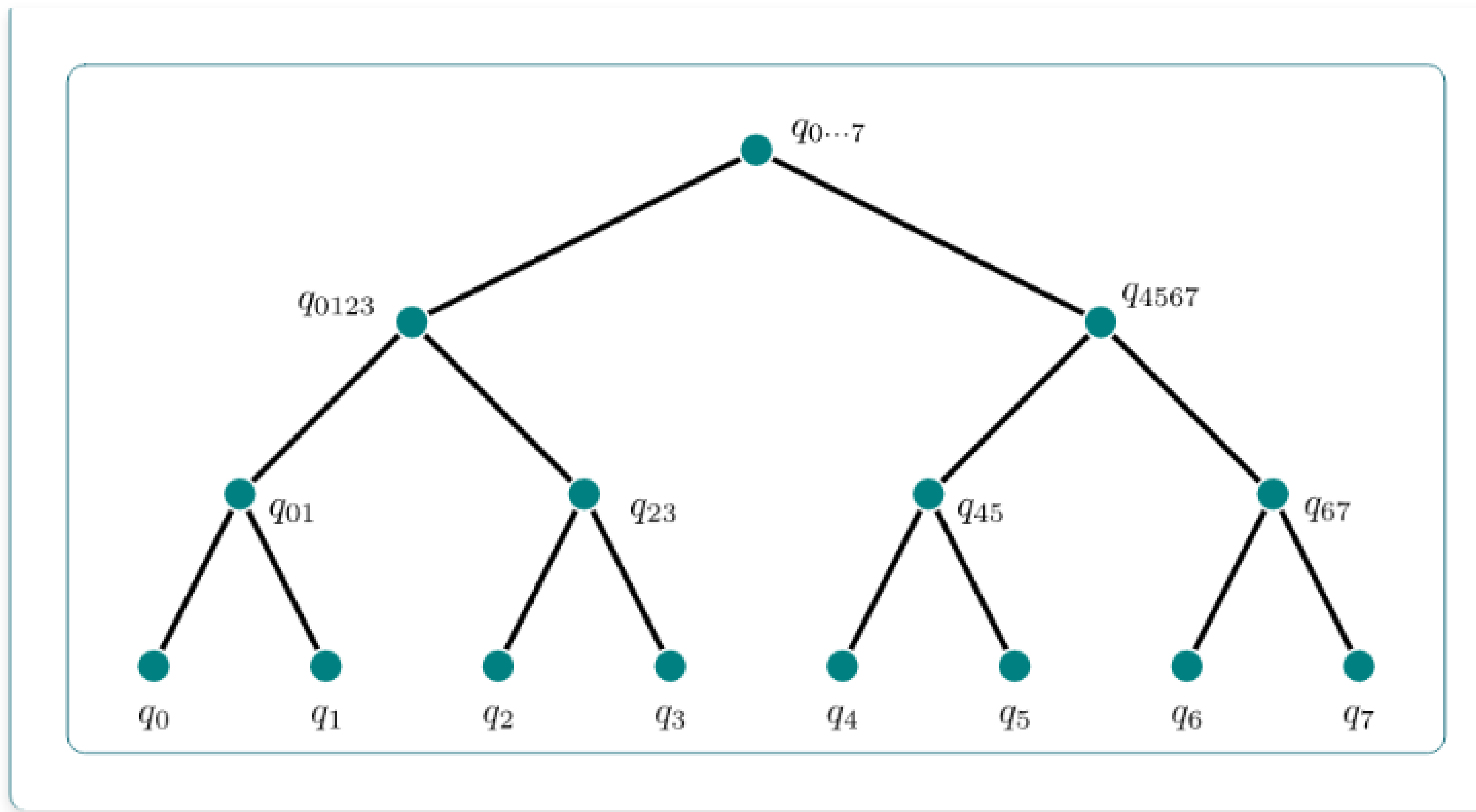
- Successful pivot in $O(\log n)$ time



- Write polygon as symmetries acting on $\vec{e} = (1, 0, 0)$
- Position of vertex n is

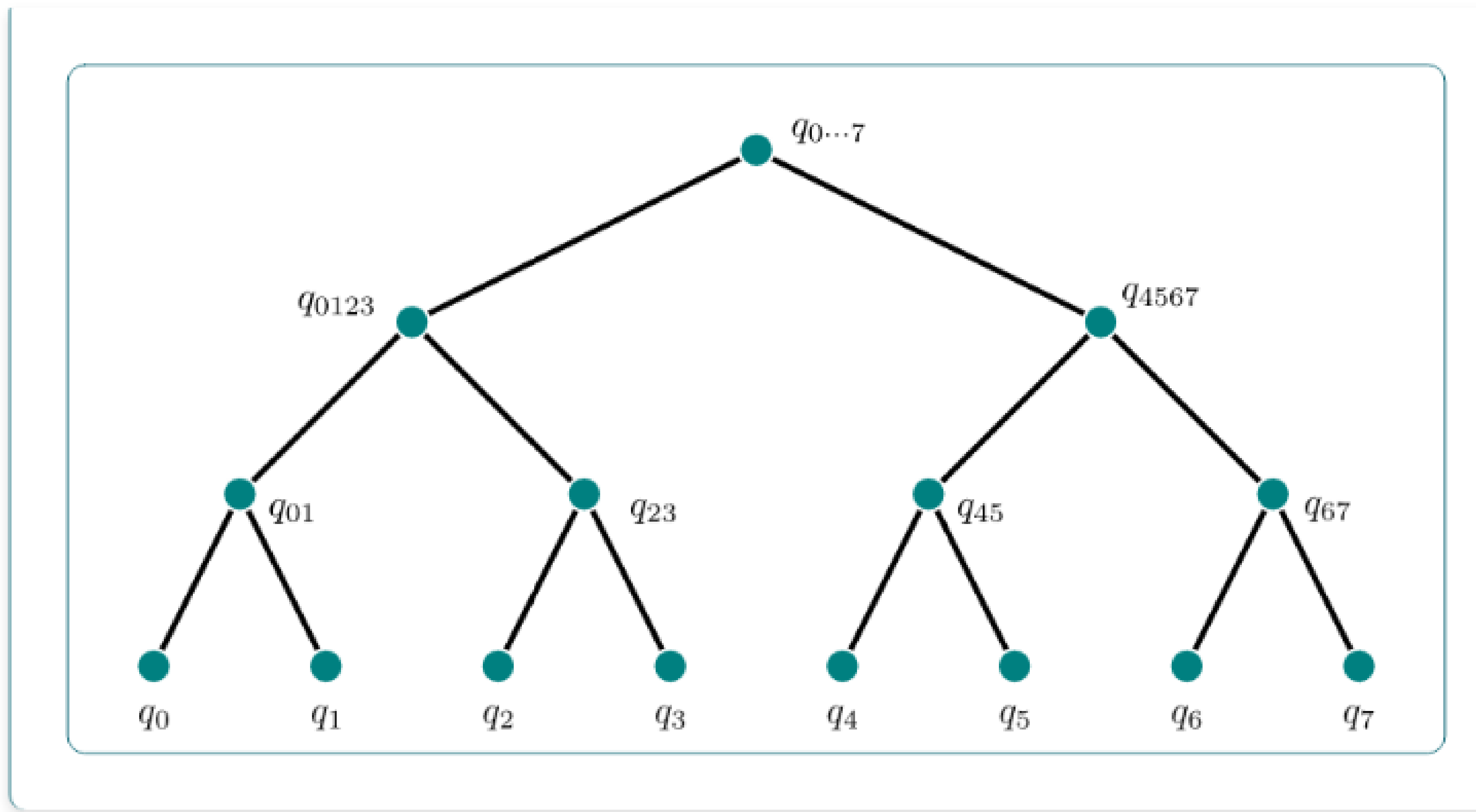
$$\vec{X}_n = \sum_{k=0}^{n-1} (q_0 q_1 \cdots q_k)\, \vec{e}$$

Store polygon in a tree



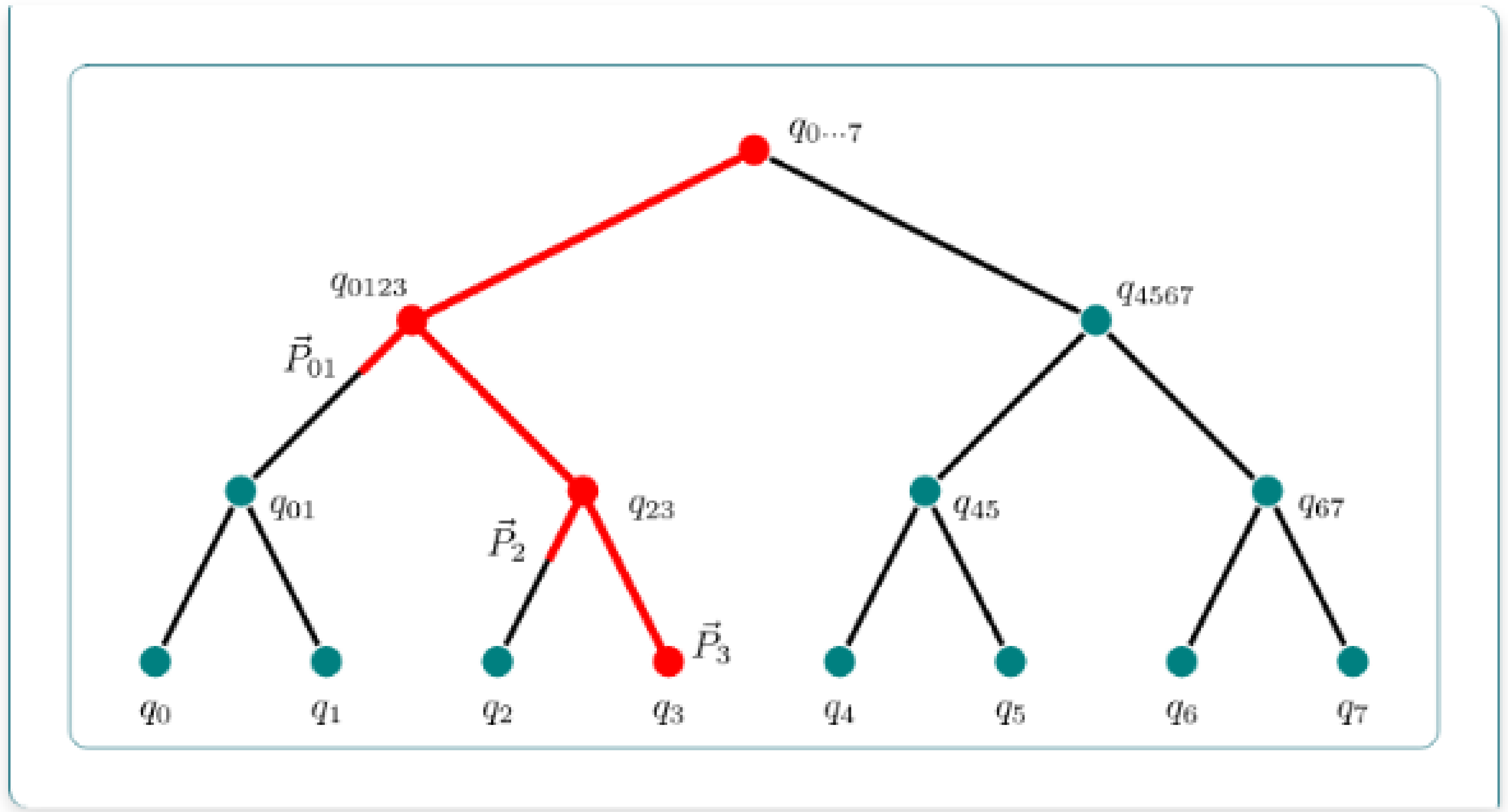
- Leaf k stores symmetry q_k and a position $\vec{P}_k = q_k\vec{e}$
- Internal nodes stores $q_n = q_\ell q_r$ and a position $\vec{P}_n = \vec{P}_\ell + q_\ell\vec{P}_r$

Store polygon in a tree



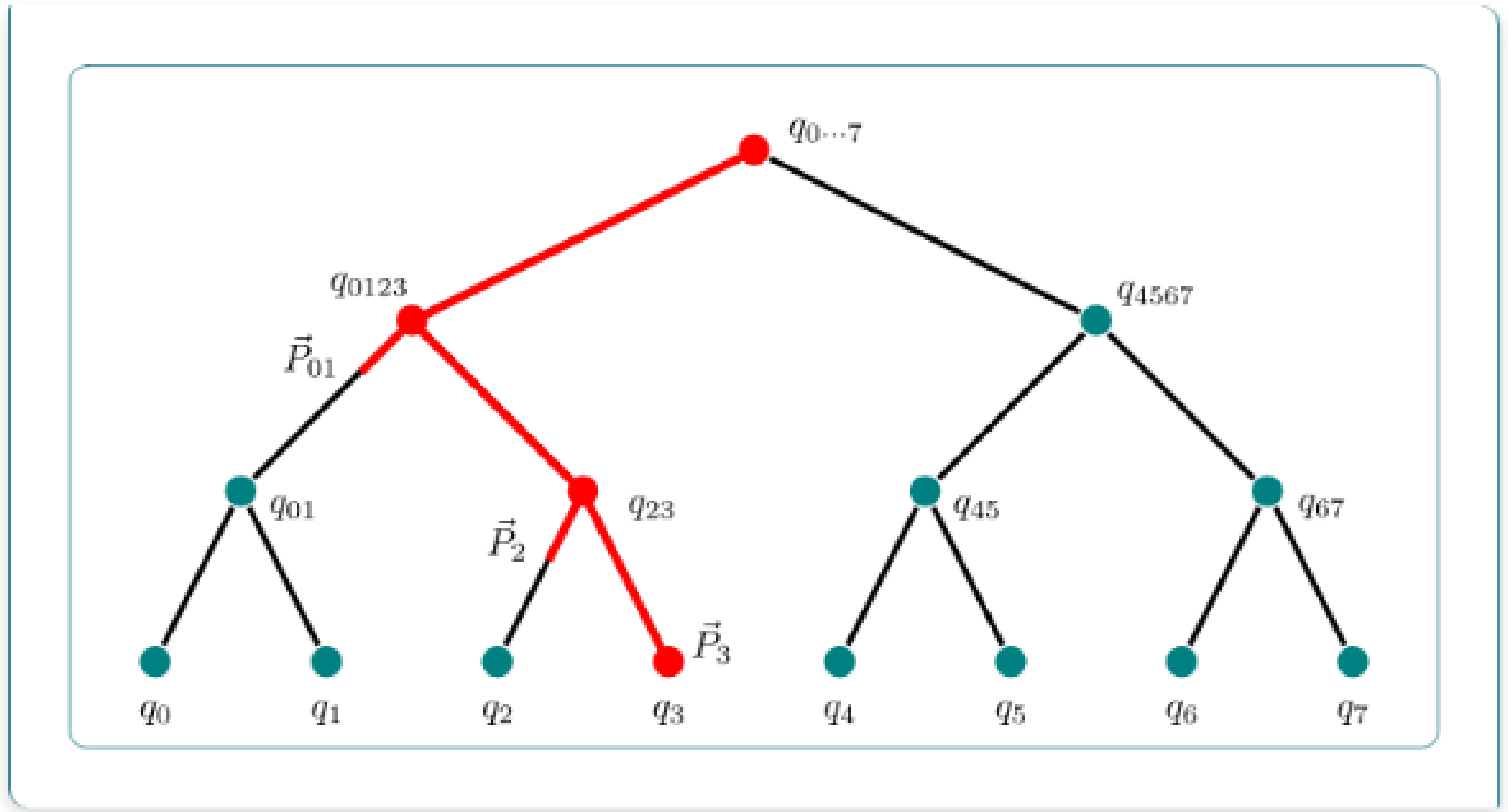
- Leaf k stores symmetry q_k and a position $\vec{P}_k = q_k \vec{e}$
- Internal nodes stores $q_n = q_\ell q_r$ and a position $\vec{P}_n = \vec{P}_\ell + q_\ell \vec{P}_r$
- Compute polygon vertex positions using $q_n, \vec{P}_n$

Compute a position



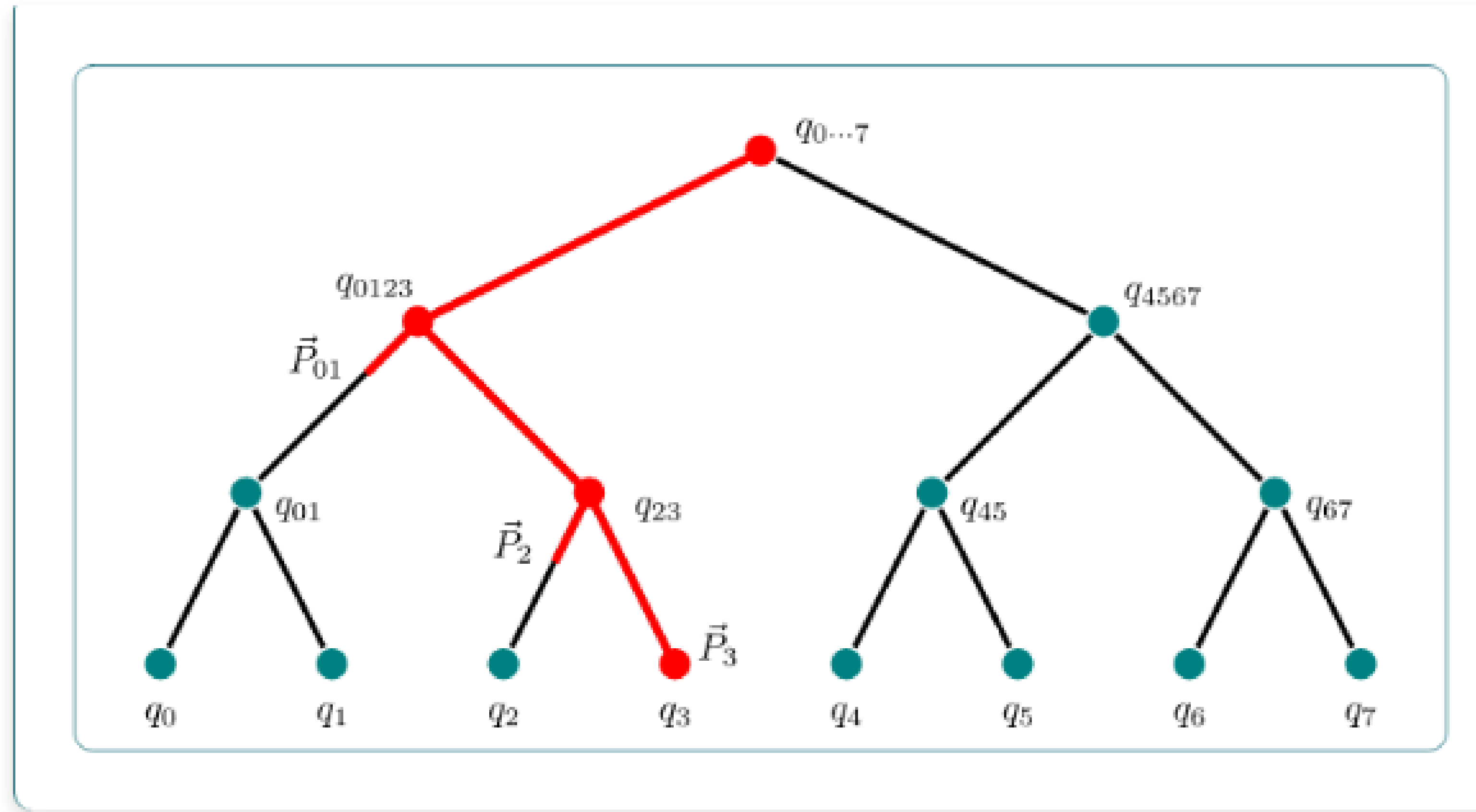
- Position of vertex 4 $\equiv$ end of 3rd polygon edge

Compute a position



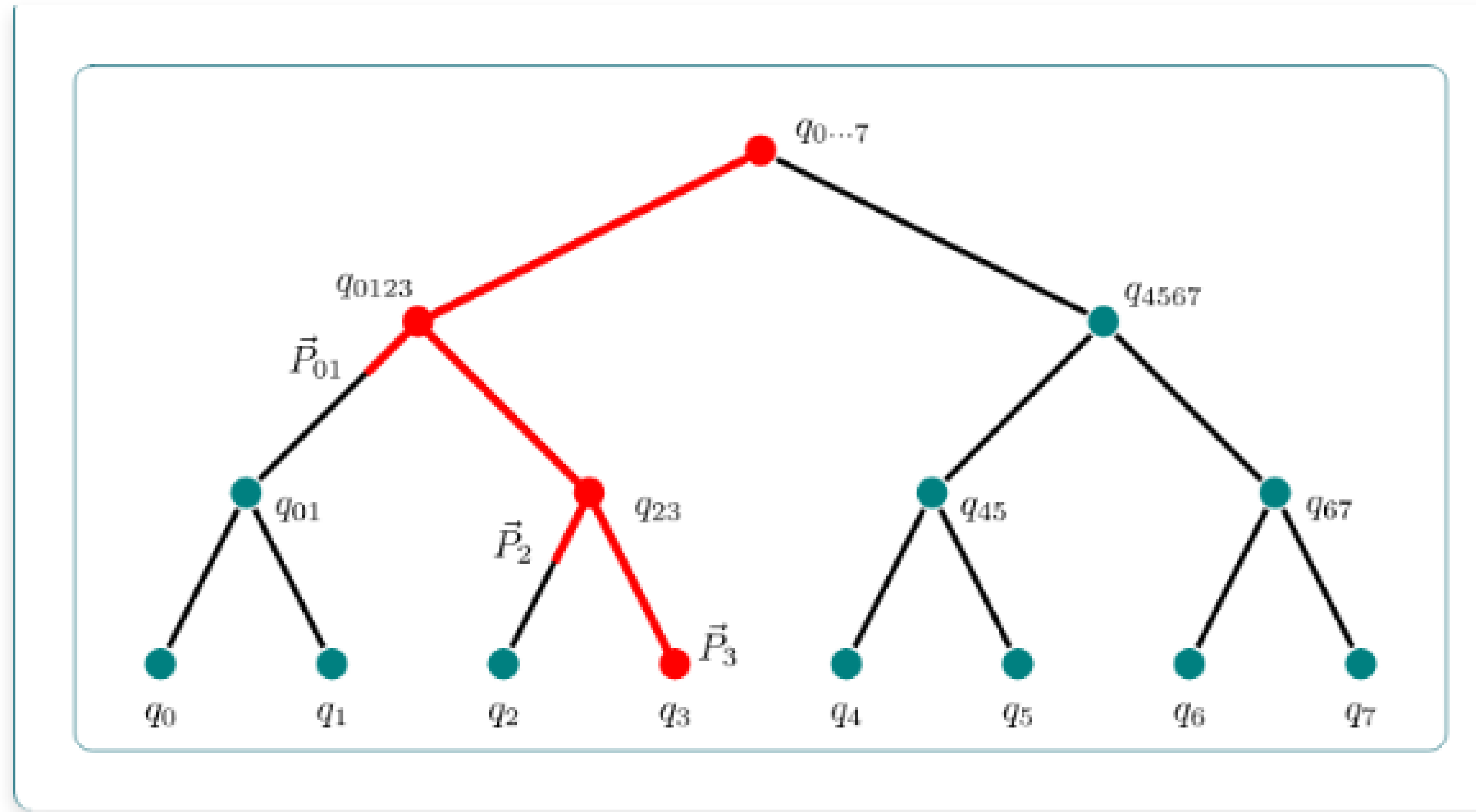
- Position of vertex 4 $\equiv$ end of 3rd polygon edge
  - $\vec{X}_4 = (q_0 + q_{01} + q_{012} + q_{0123})\,\vec{e}$

Compute a position



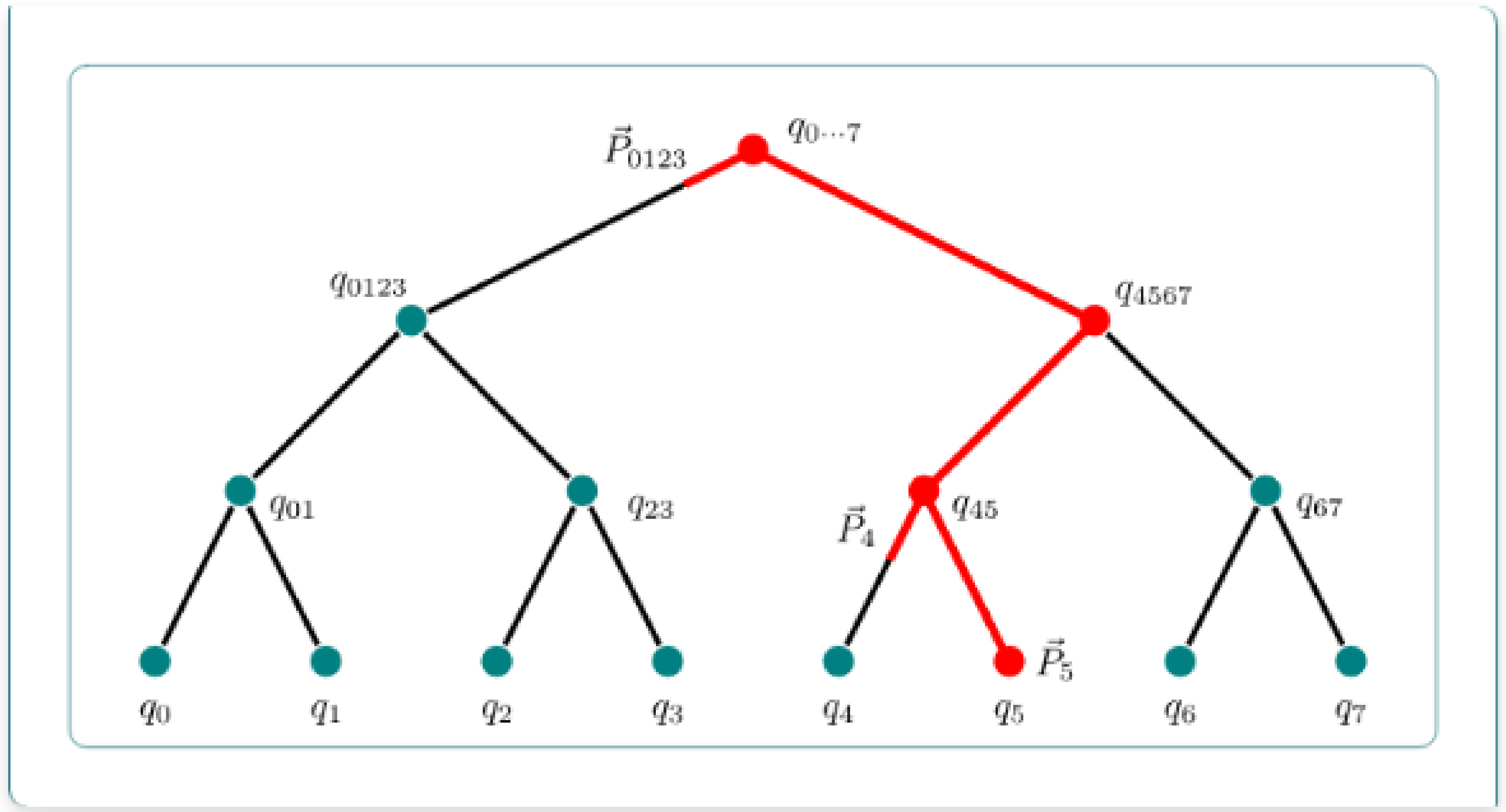
- Position of vertex 4 $\equiv$ end of 3rd polygon edge
  - $\vec{X}_4 = (q_0 + q_{01} + q_{012} + q_{0123})\,\vec{e}$
  - $\vec{X}_4 = (q_0 + q_{01})\,\vec{e} + q_{01}\,(q_2\vec{e} + q_2(q_3\vec{e}))$

Compute a position



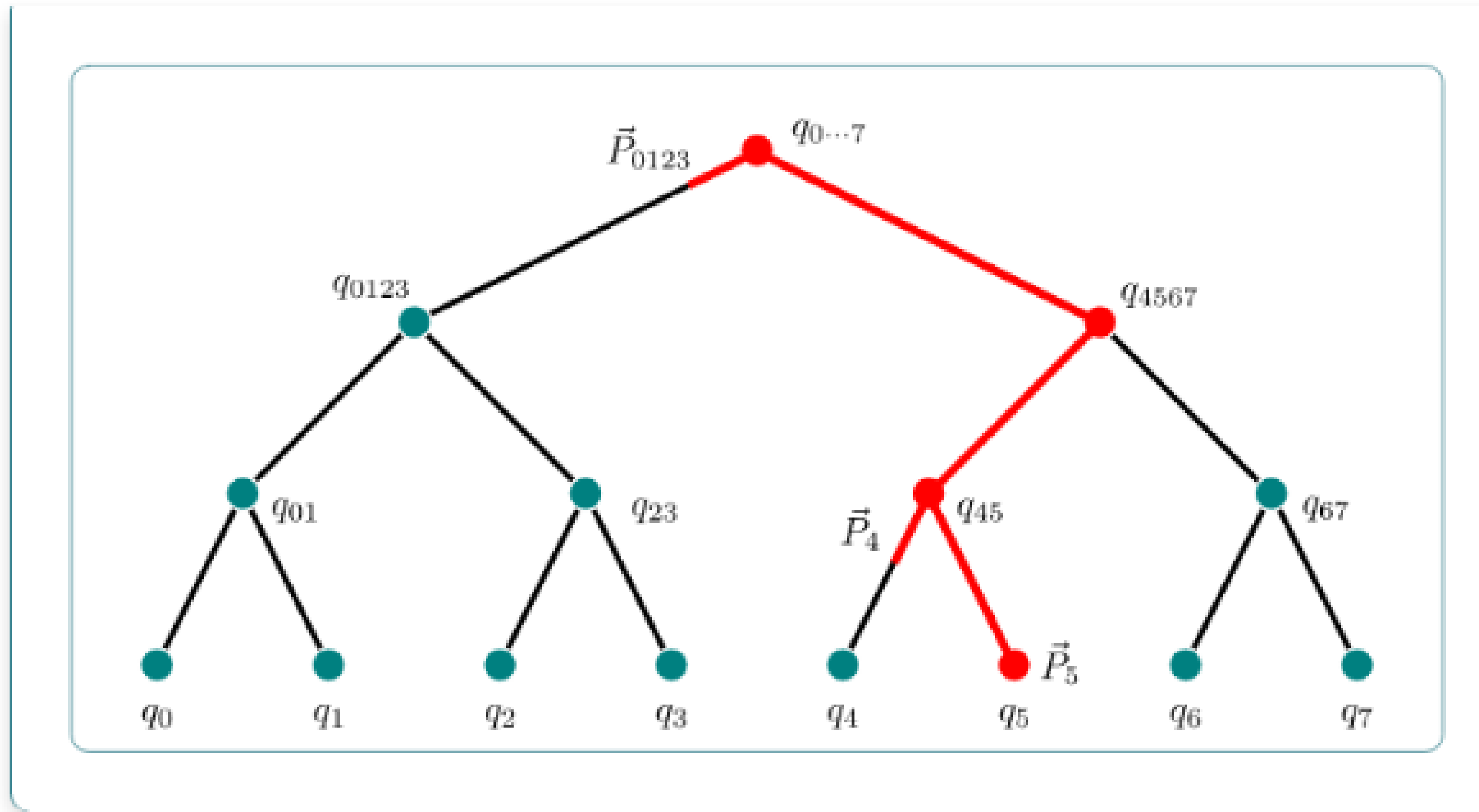
- Position of vertex 4 $\equiv$ end of 3rd polygon edge
  - $\vec{X}_4 = (q_0 + q_{01} + q_{012} + q_{0123})\,\vec{e}$
  - $\vec{X}_4 = (q_0 + q_{01})\,\vec{e} + q_{01}\,(q_2\vec{e} + q_2(q_3\vec{e}))$
  - $\vec{X}_4 = \vec{P}_{01} + q_{01}\left(\vec{P}_2 + (q_2\vec{P}_3)\right)$
- Already computed q_{01} and P_{01}, P_2, P_3.
- Requires $O(\text{tree-depth}) = O(\log n)$ operations

Compute another position



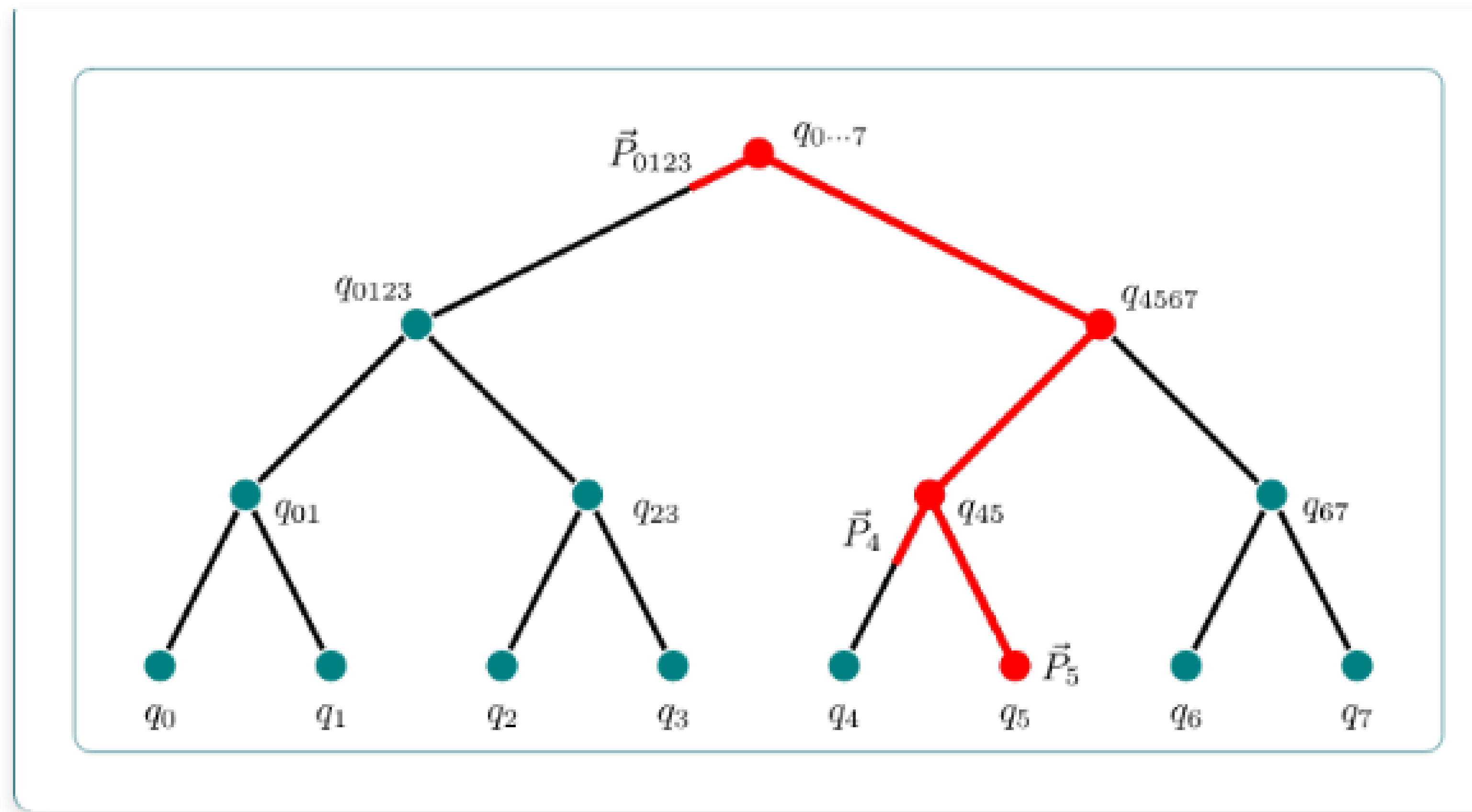
- Position of vertex 6 $\equiv$ end of 5th polygon edge

Compute another position



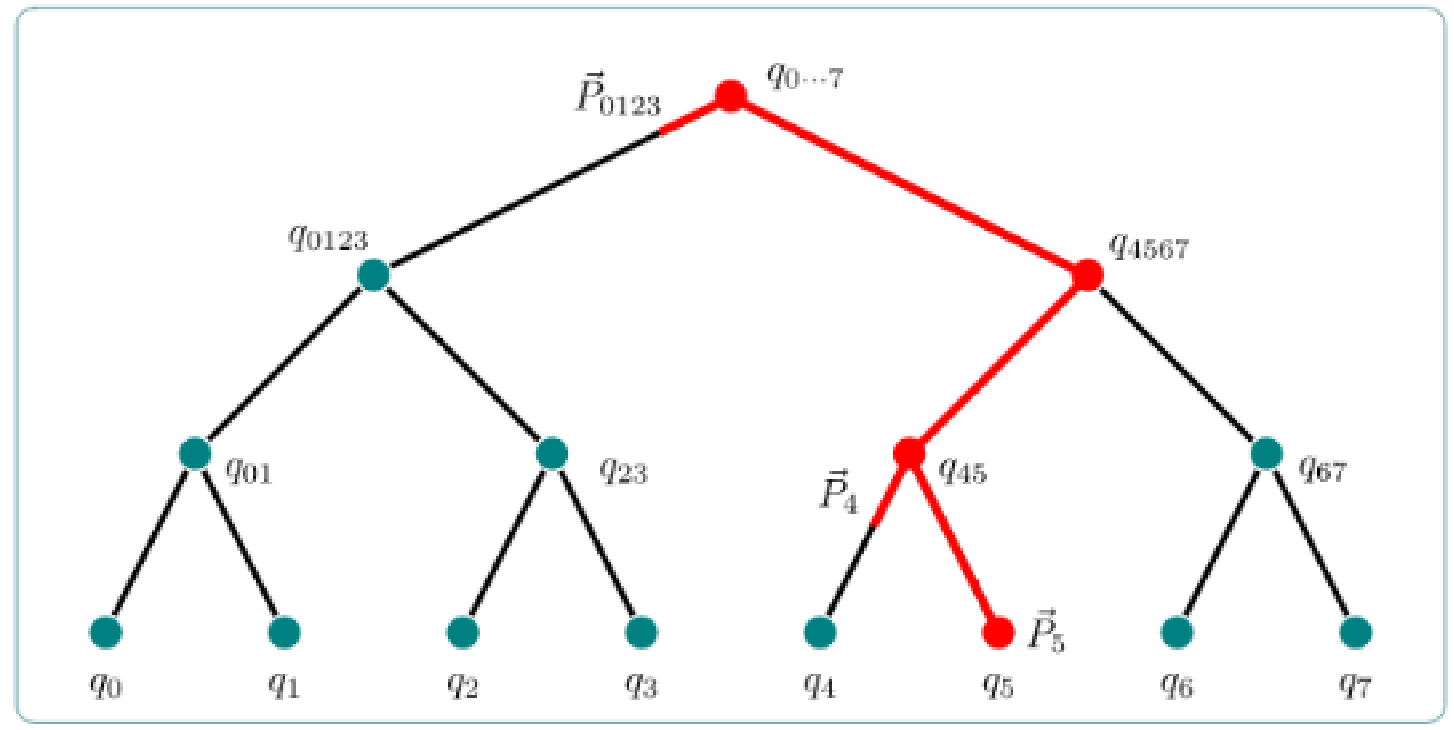
- Position of vertex 6 $\equiv$ end of 5th polygon edge
  - $\vec{X}_6 = (q_0 + q_{01} + q_{012} + q_{0123} + q_{01234} + q_{012345})\,\vec{e}$

Compute another position



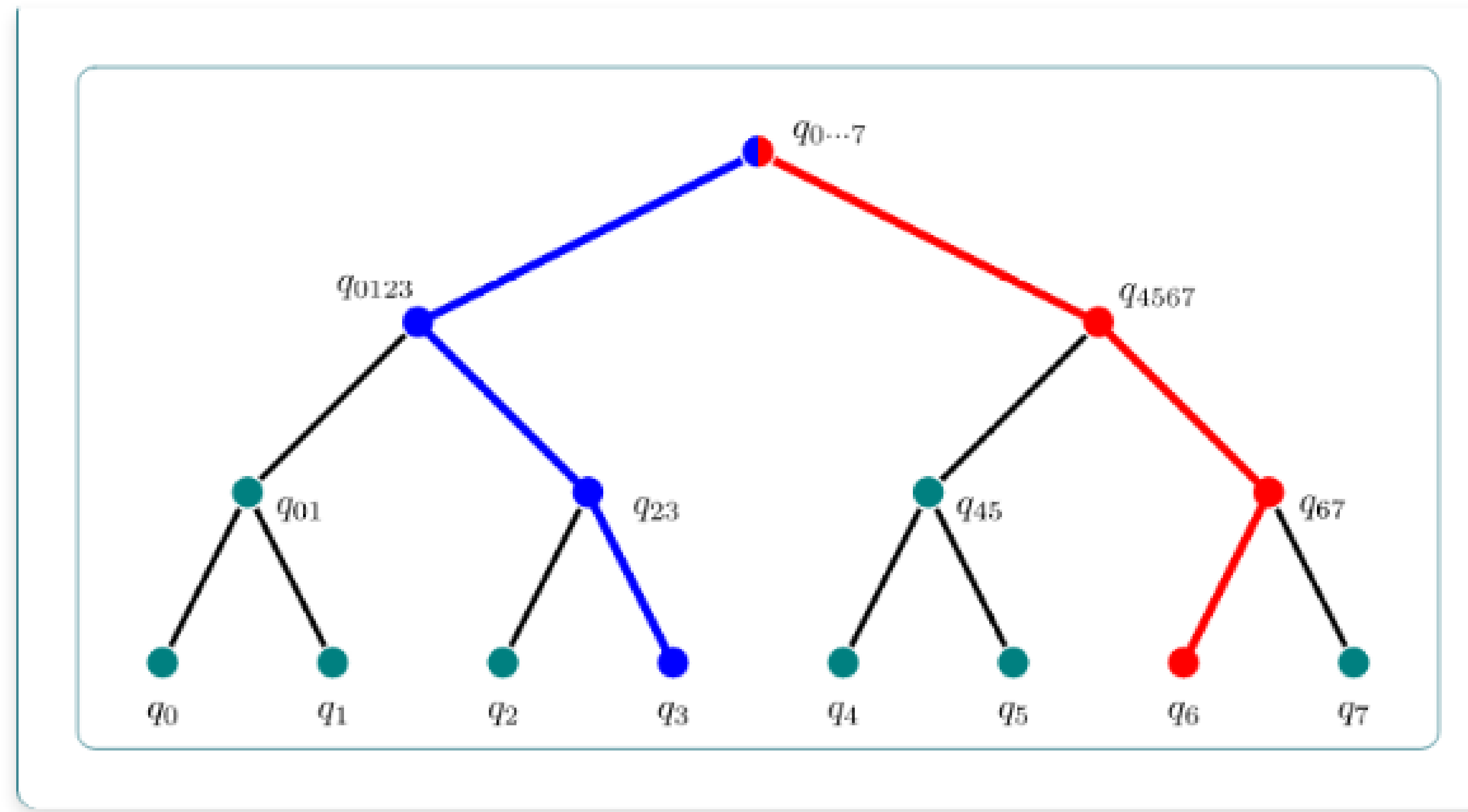
- Position of vertex 6 $\equiv$ end of 5th polygon edge
  - $\vec{X}_6 = (q_0 + q_{01} + q_{012} + q_{0123} + q_{01234} + q_{012345})\,\vec{e}$
  - $\vec{X}_6 = (q_0 + q_{01} + q_{012} + q_{0123})\,\vec{e} + q_{0123}\,(q_4\vec{e} + q_4(q_5\vec{e}))$

Compute another position



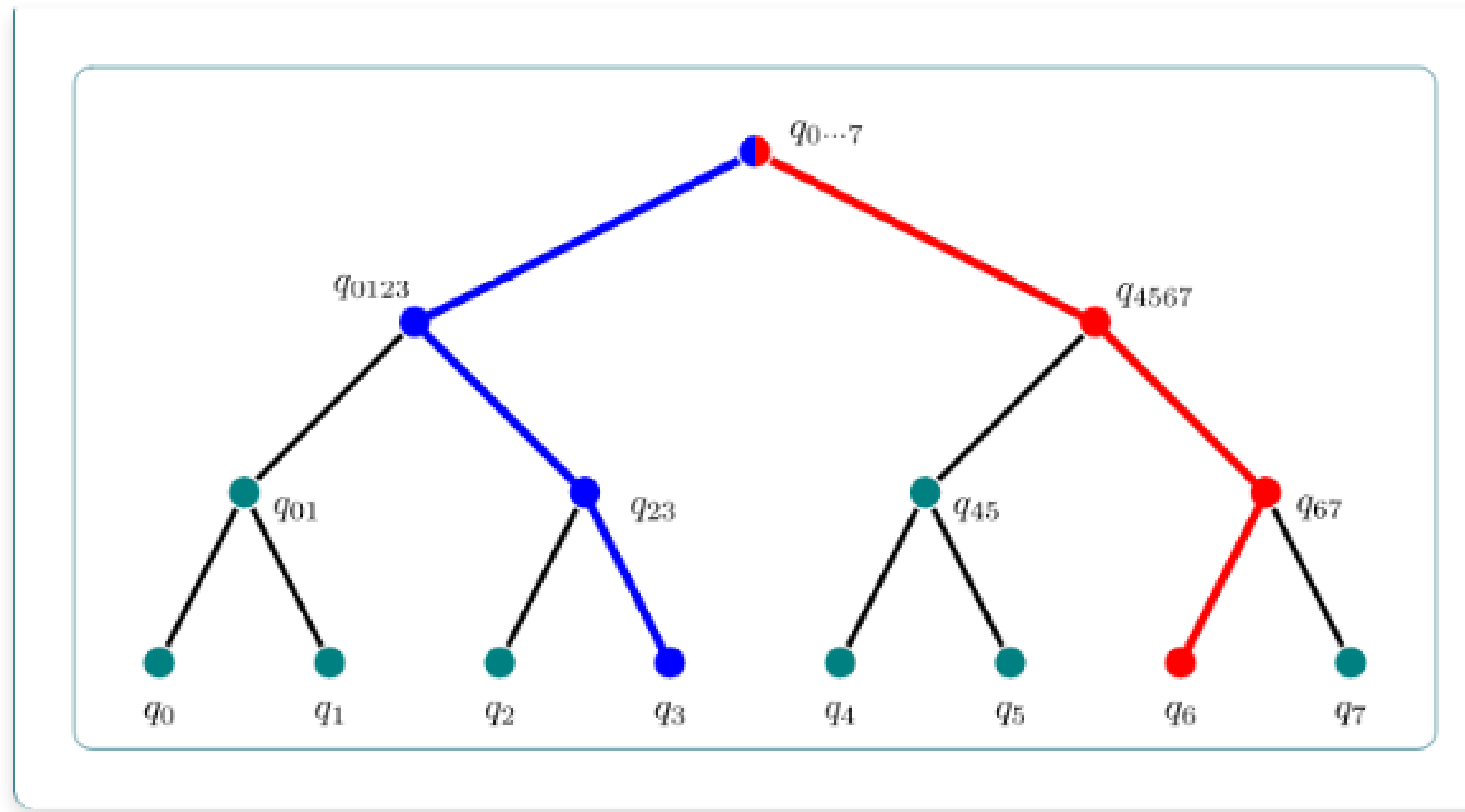
- Position of vertex 6 $\equiv$ end of 5th polygon edge
  - $\vec{X}_6 = (q_0 + q_{01} + q_{012} + q_{0123} + q_{01234} + q_{012345})\,\vec{e}$
  - $\vec{X}_6 = (q_0 + q_{01} + q_{012} + q_{0123})\,\vec{e} + q_{0123}\,(q_4\vec{e} + q_4(q_5\vec{e}))$
  - $\vec{X}_6 = \vec{P}_{0123} + q_{0123}\left(\vec{P}_4 + q_4\overrightarrow{P_5}\right)$
- Already computed q_{0123} and P_{0123}, P_4, P_5.
- Requires $O(\text{tree-depth}) = O(\log n)$ operations

Update after pivot at vertices 3 and 6



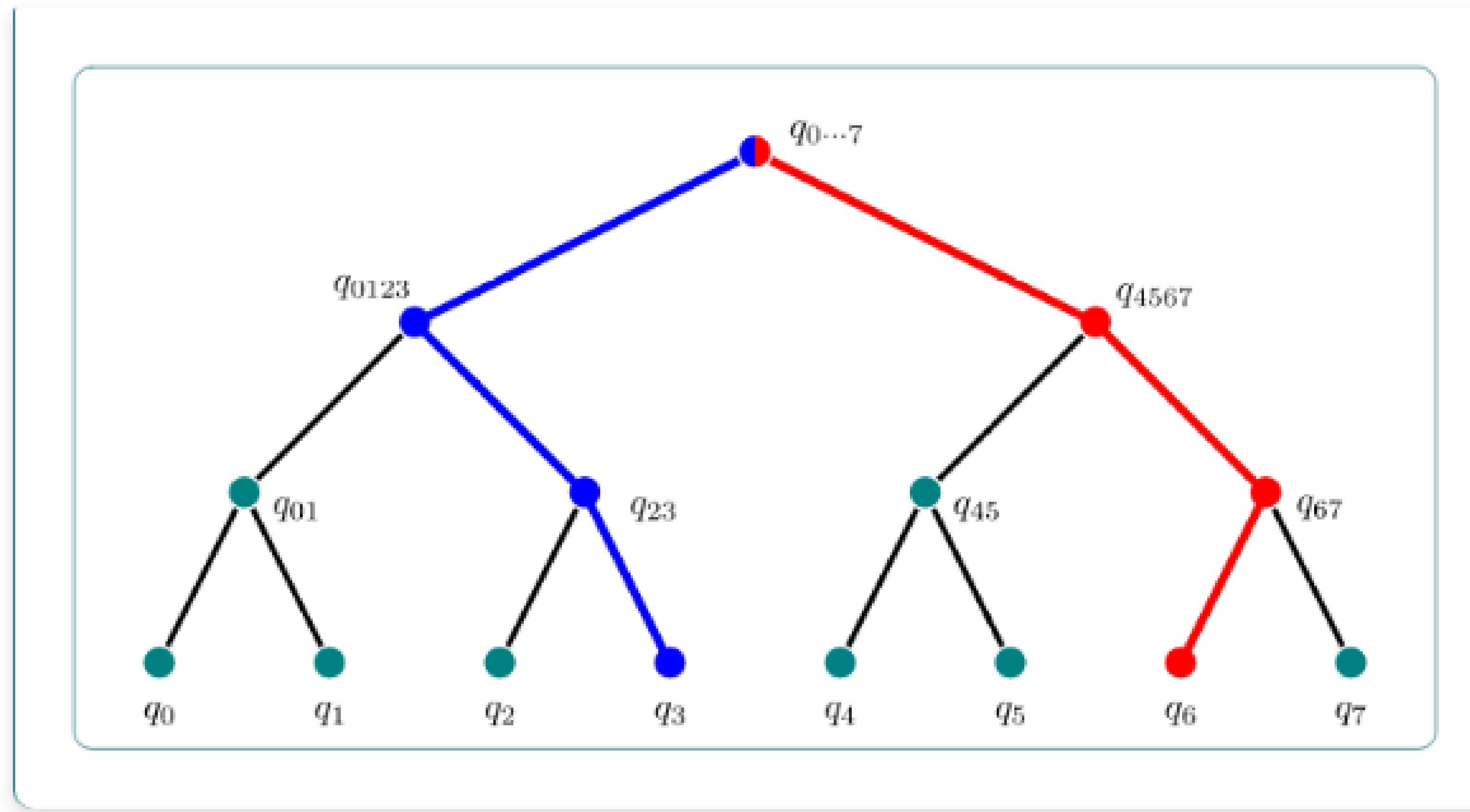
- Move up the tree from leaves 3 and 6
- Recomputing data at each node requires $O(\text{tree-depth}) = O(\log n)$ operations

Update after pivot at vertices 3 and 6



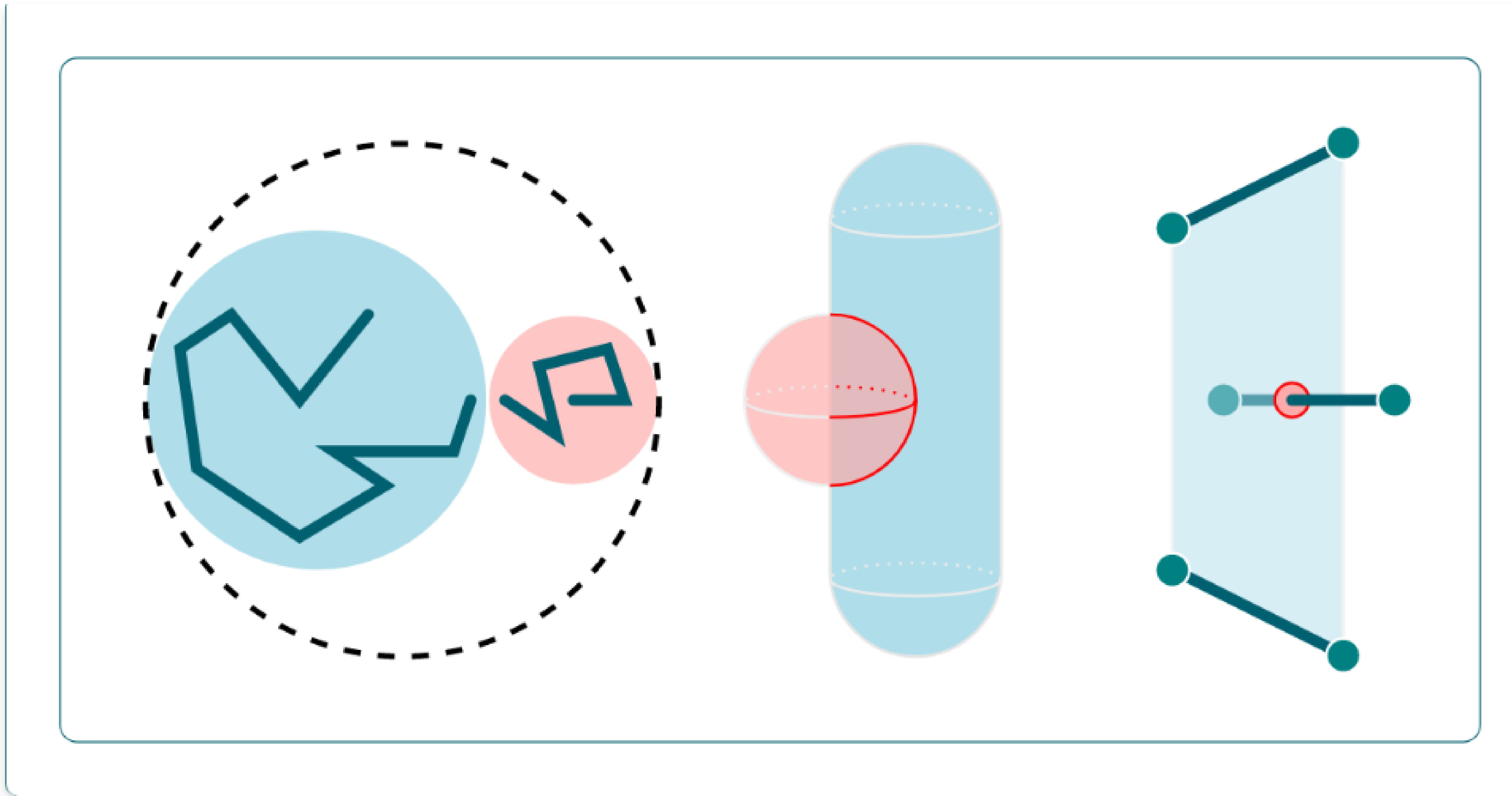
- Move up the tree from leaves 3 and 6
- Recomputing data at each node requires $O(\text{tree-depth}) = O(\log n)$ operations
- Very fast Markov chain sampling of ERP (no topology checks)
  - Auto-correlation time for $R_g(n) \approx O(\log n)$
  - ERP sampling in sublinear time
  - Takes longer to write down than to sample!

Update after pivot at vertices 3 and 6



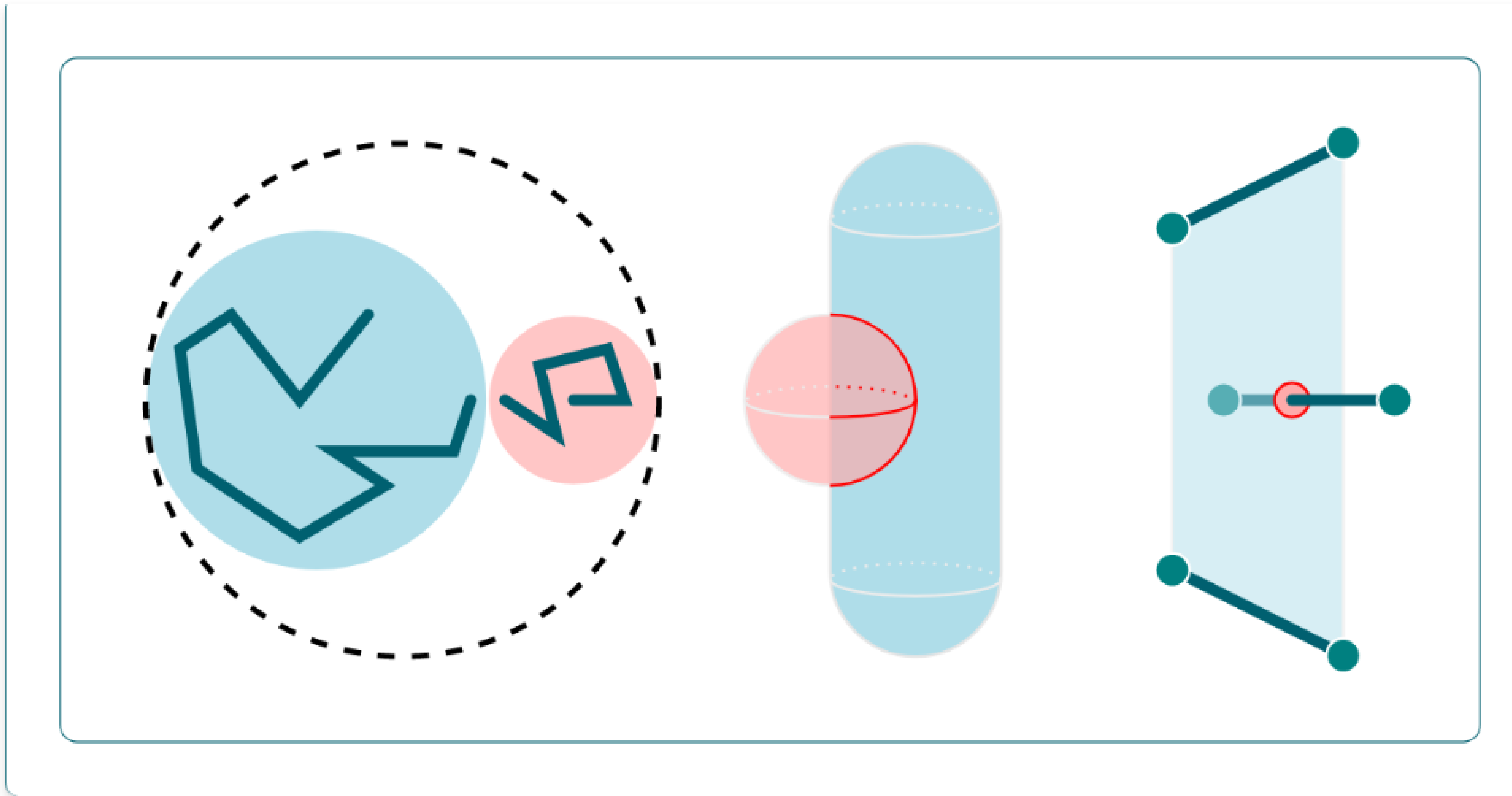
- Move up the tree from leaves 3 and 6
- Recomputing data at each node requires $O(\text{tree-depth}) = O(\log n)$ operations
- Very fast Markov chain sampling of ERP (no topology checks)
  - Auto-correlation time for $R_g(n) \approx O(\log n)$
  - ERP sampling in sublinear time
  - Takes longer to write down than to sample!
- Harder on lattice — must pick pairs carefully to stay on lattice

For topology checks



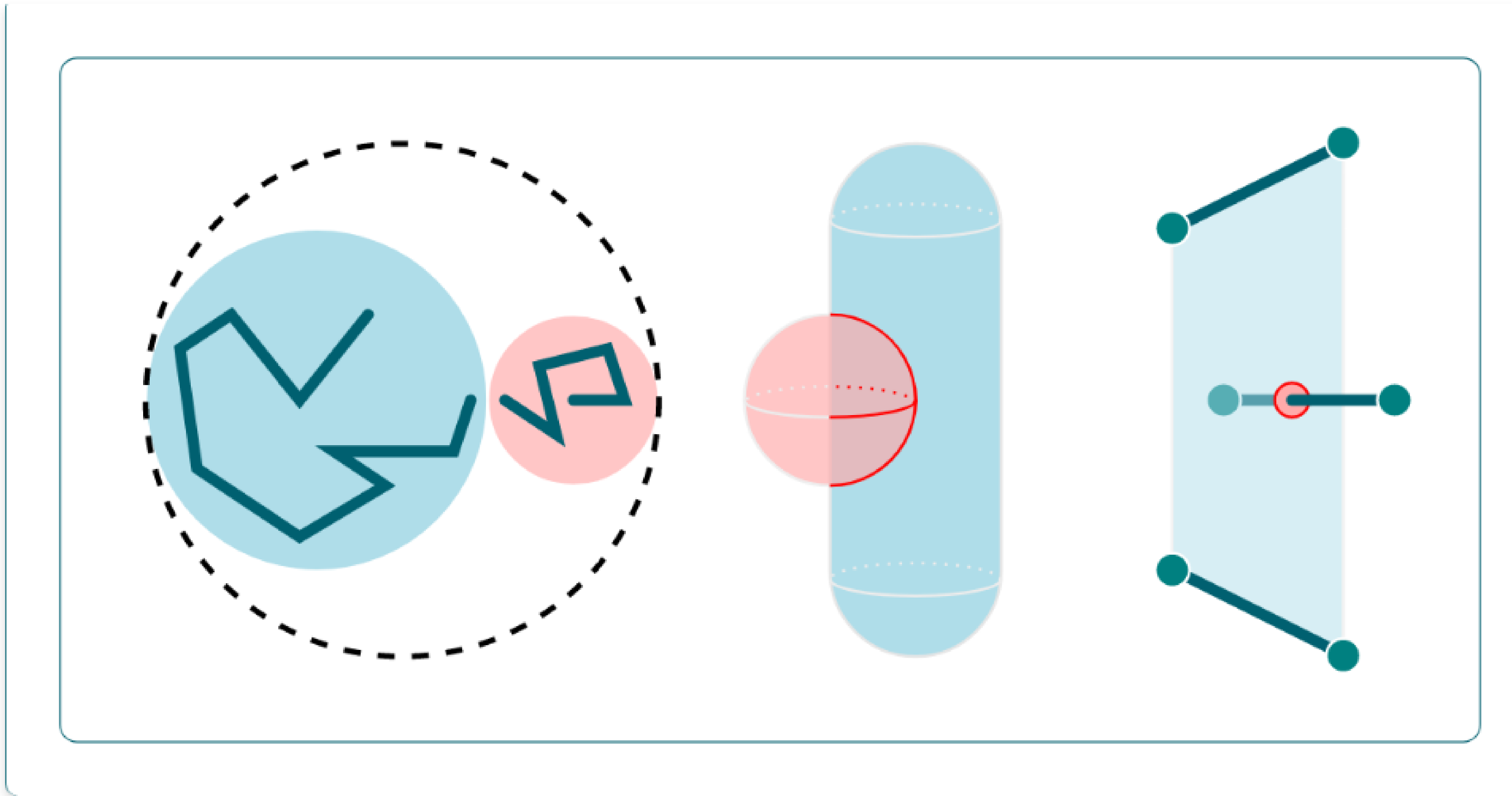
- Lots of basic vector and quaternion manipulation

For topology checks



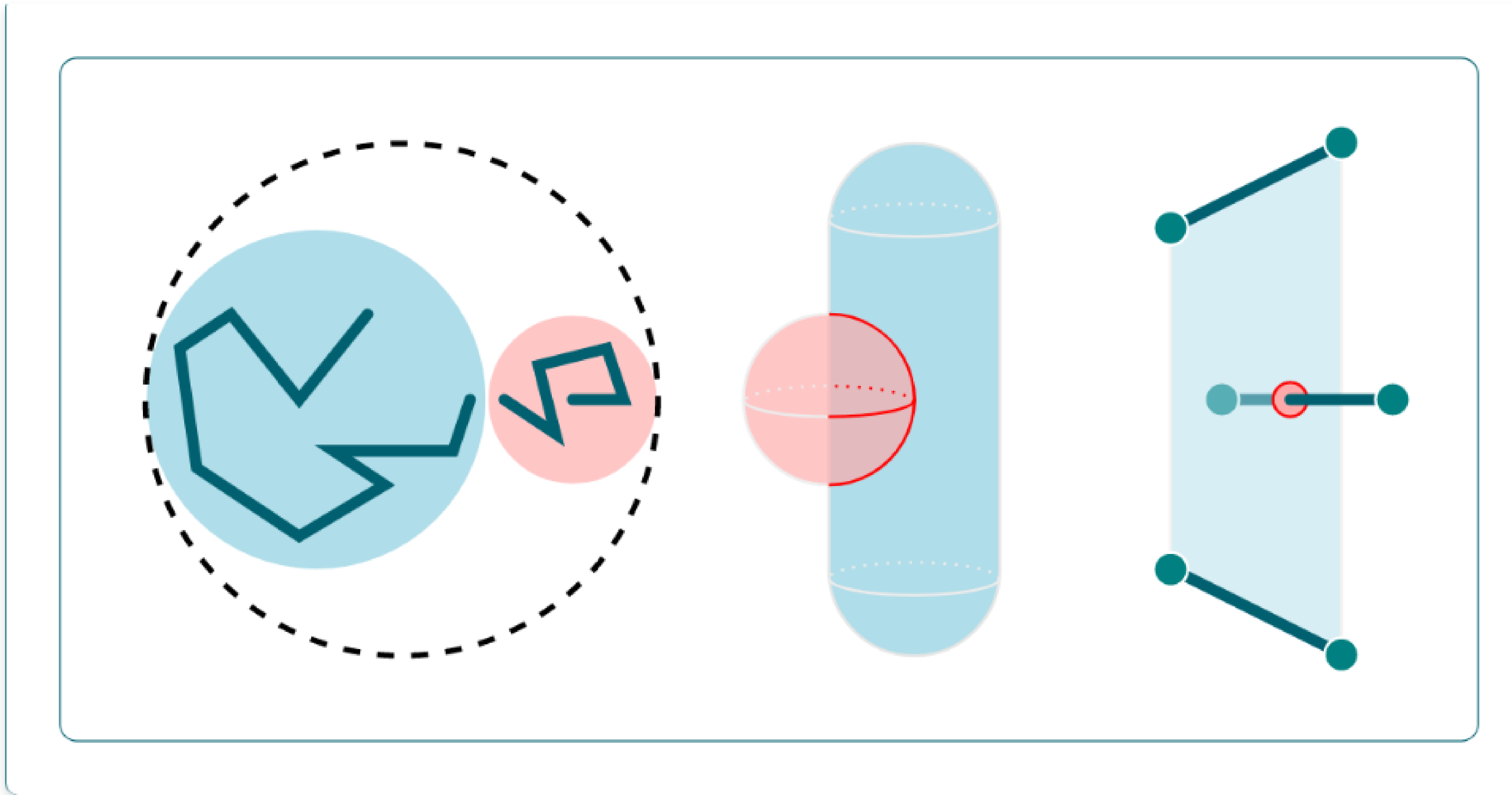
- Lots of basic vector and quaternion manipulation
- Bounding sphere construction

For topology checks



- Lots of basic vector and quaternion manipulation
- Bounding sphere construction
- Sphere intersects sphere-capped-cylinder test

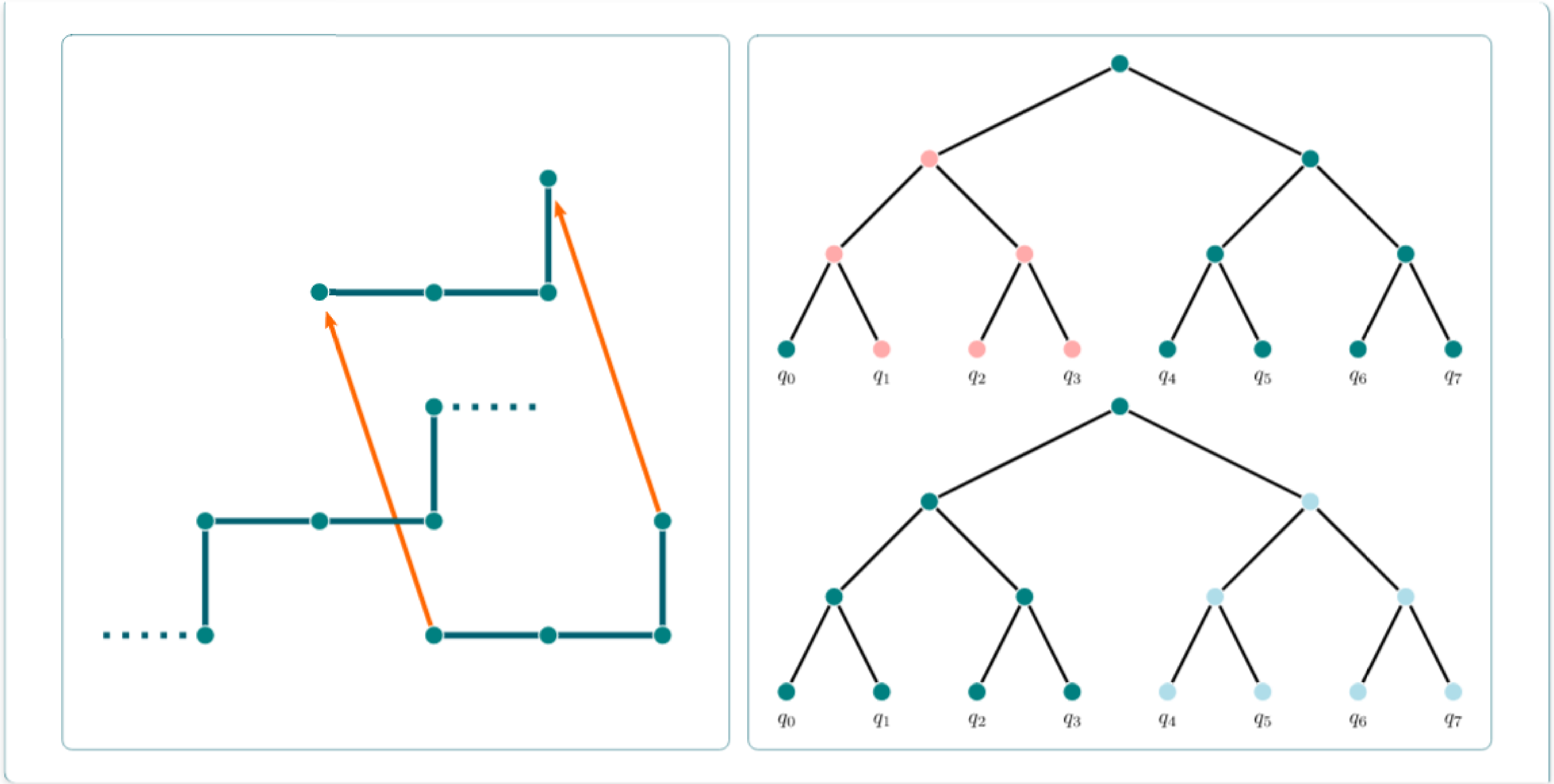
For topology checks



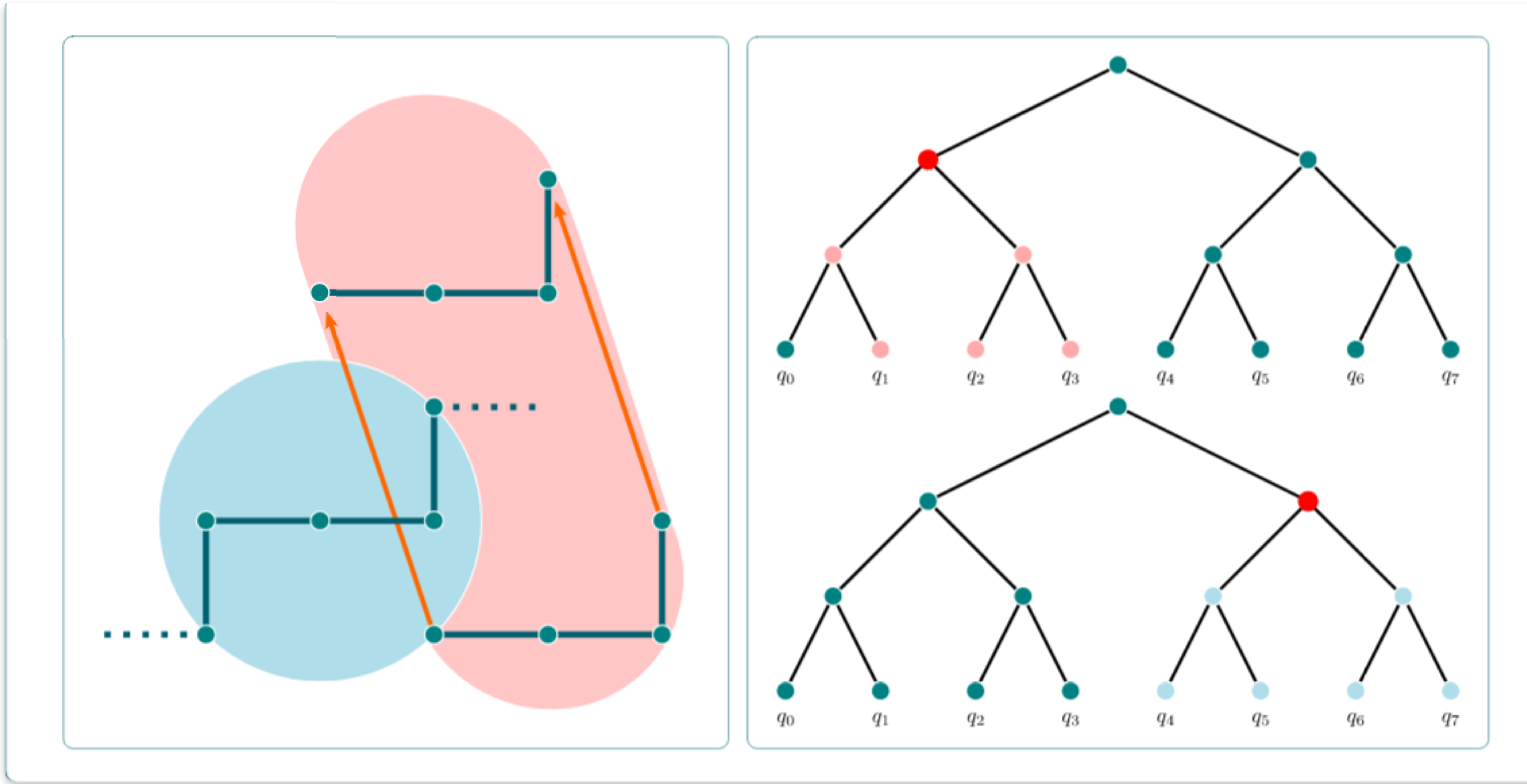
- Lots of basic vector and quaternion manipulation
- Bounding sphere construction
- Sphere intersects sphere-capped-cylinder test
- Segment intersects quadrilateral test $\equiv$ Möller-Trumbore

Fast intersection checks via bounding sphere refinements

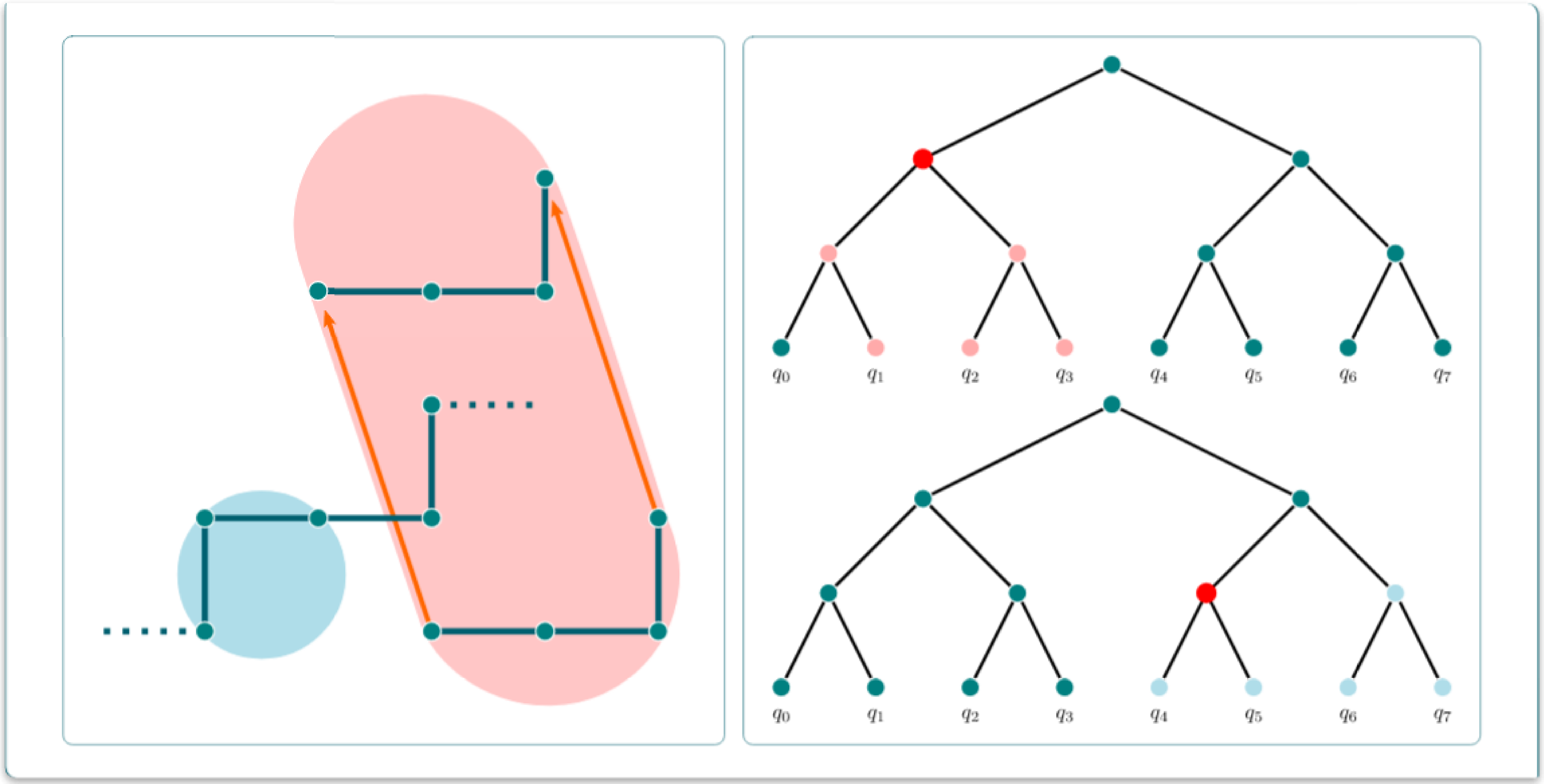
Fast intersection checks via bounding sphere refinements



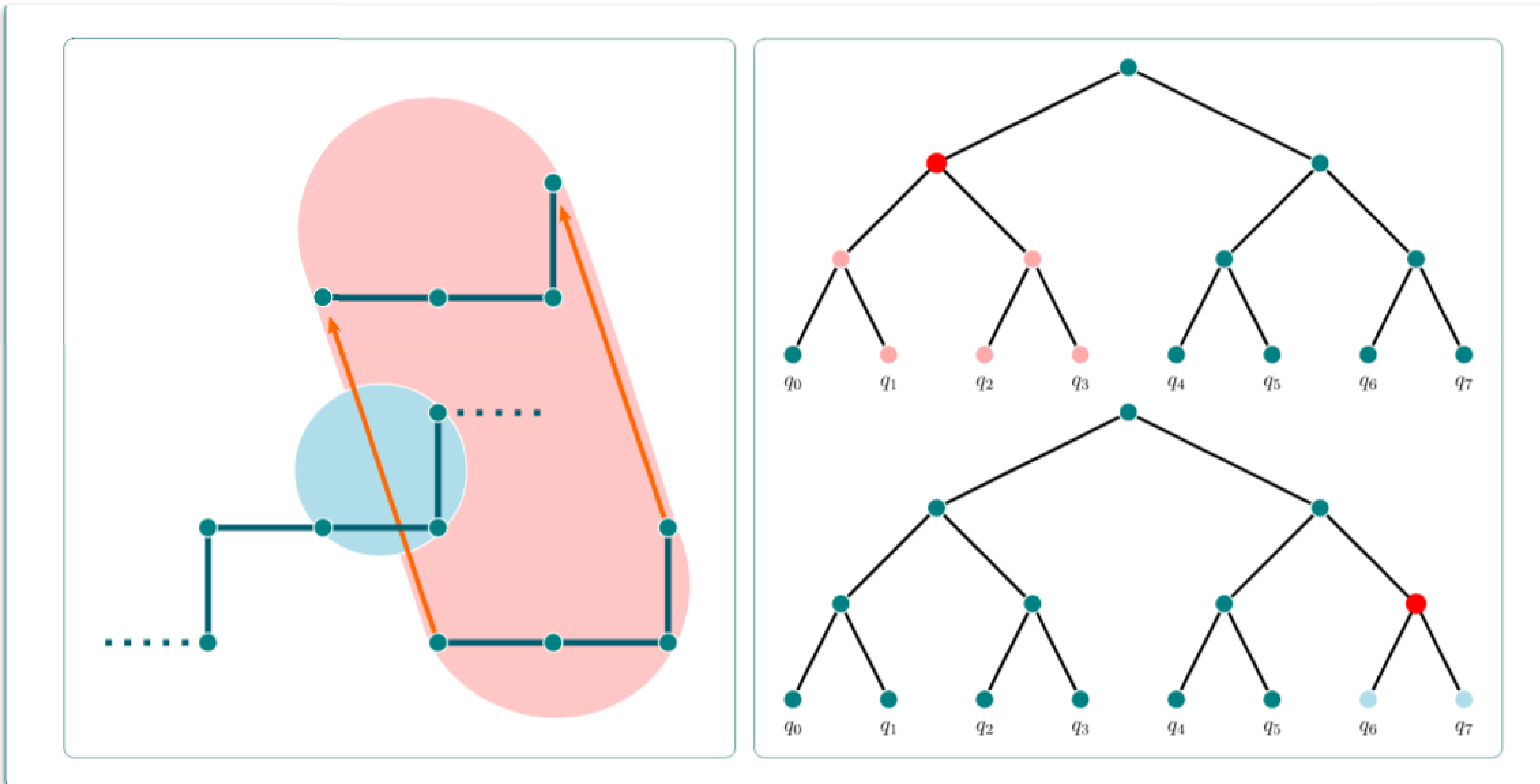
Fast intersection checks via bounding sphere refinements



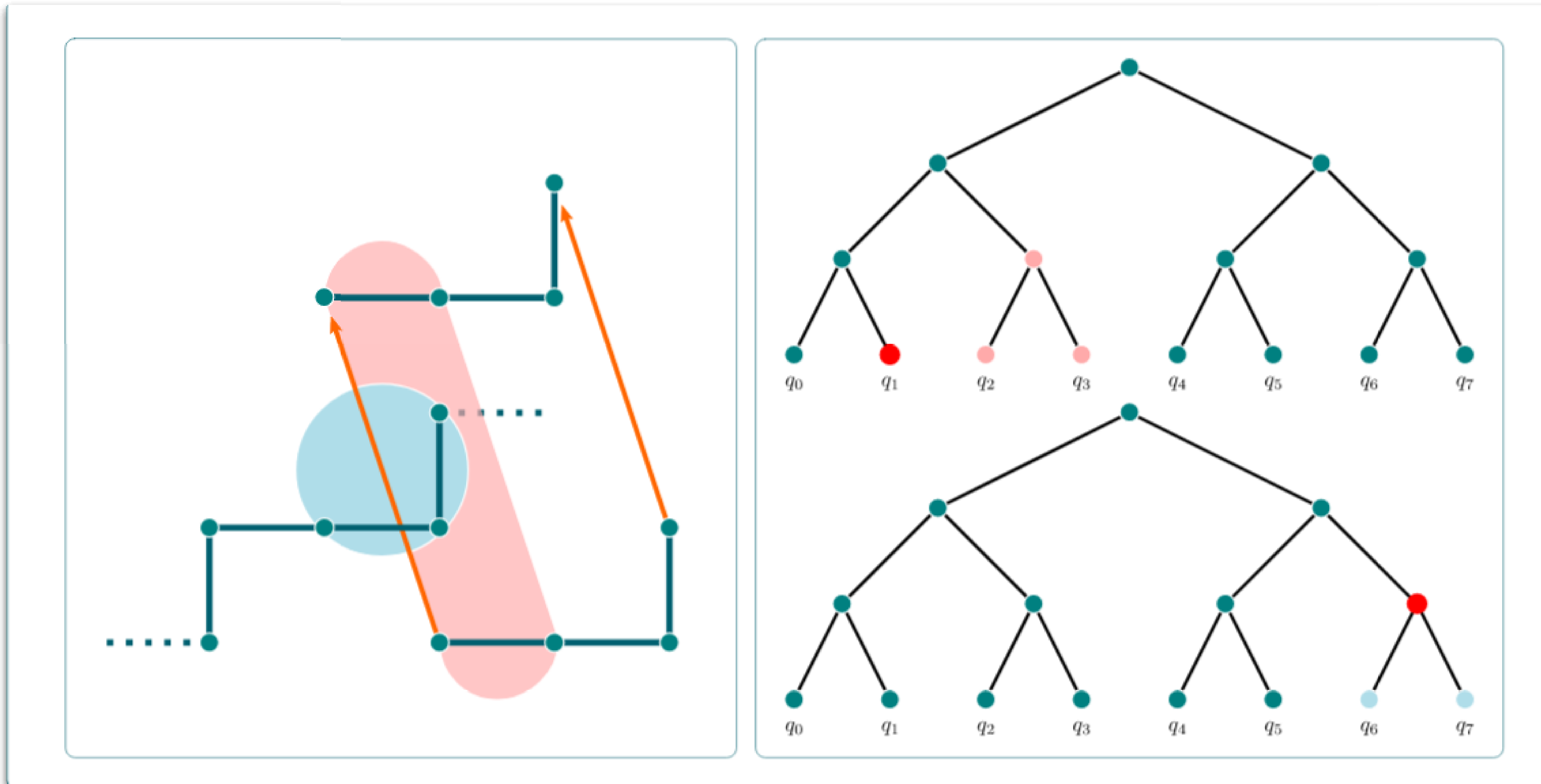
Fast intersection checks via bounding sphere refinements



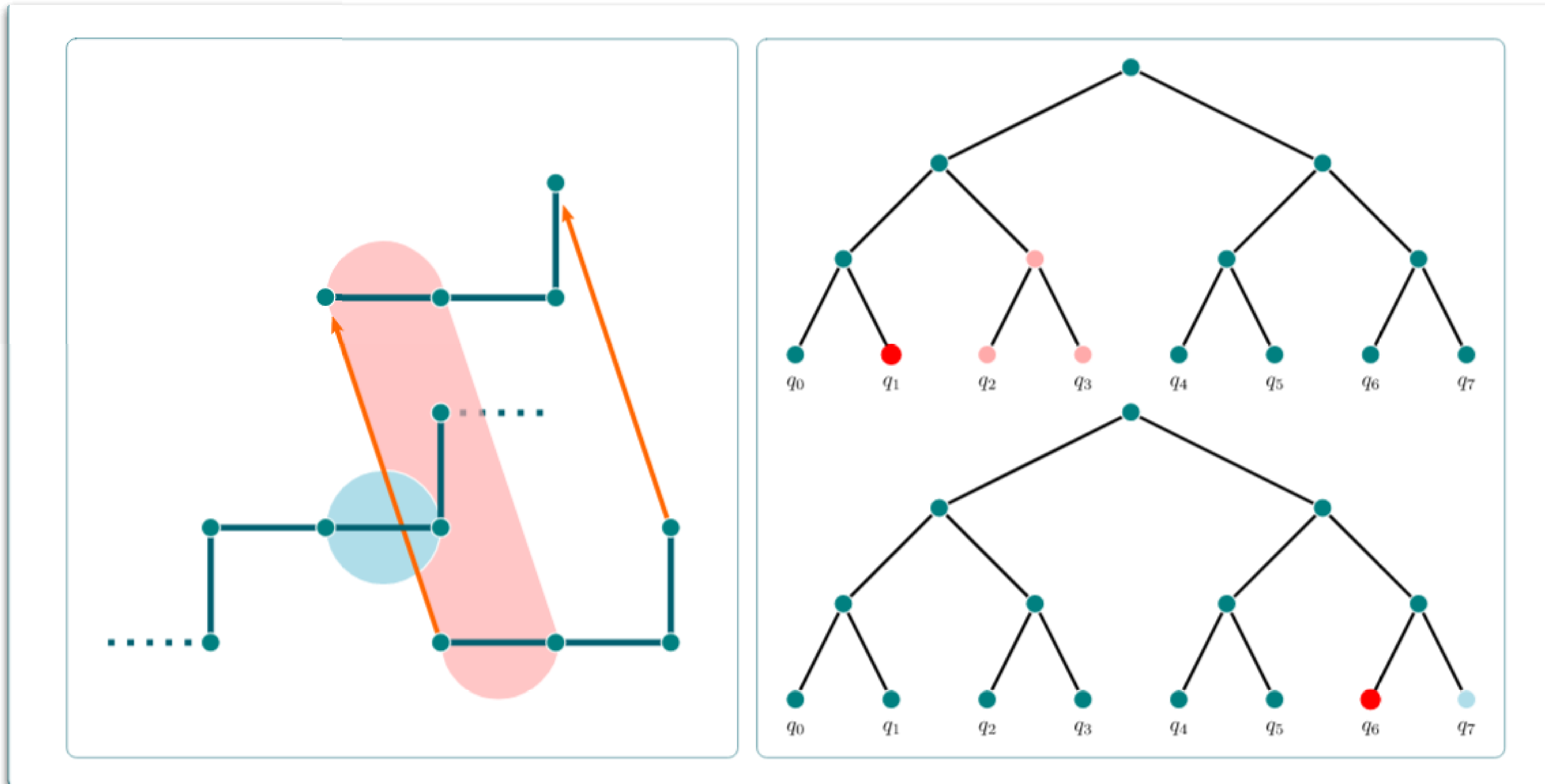
Fast intersection checks via bounding sphere refinements



Fast intersection checks via bounding sphere refinements

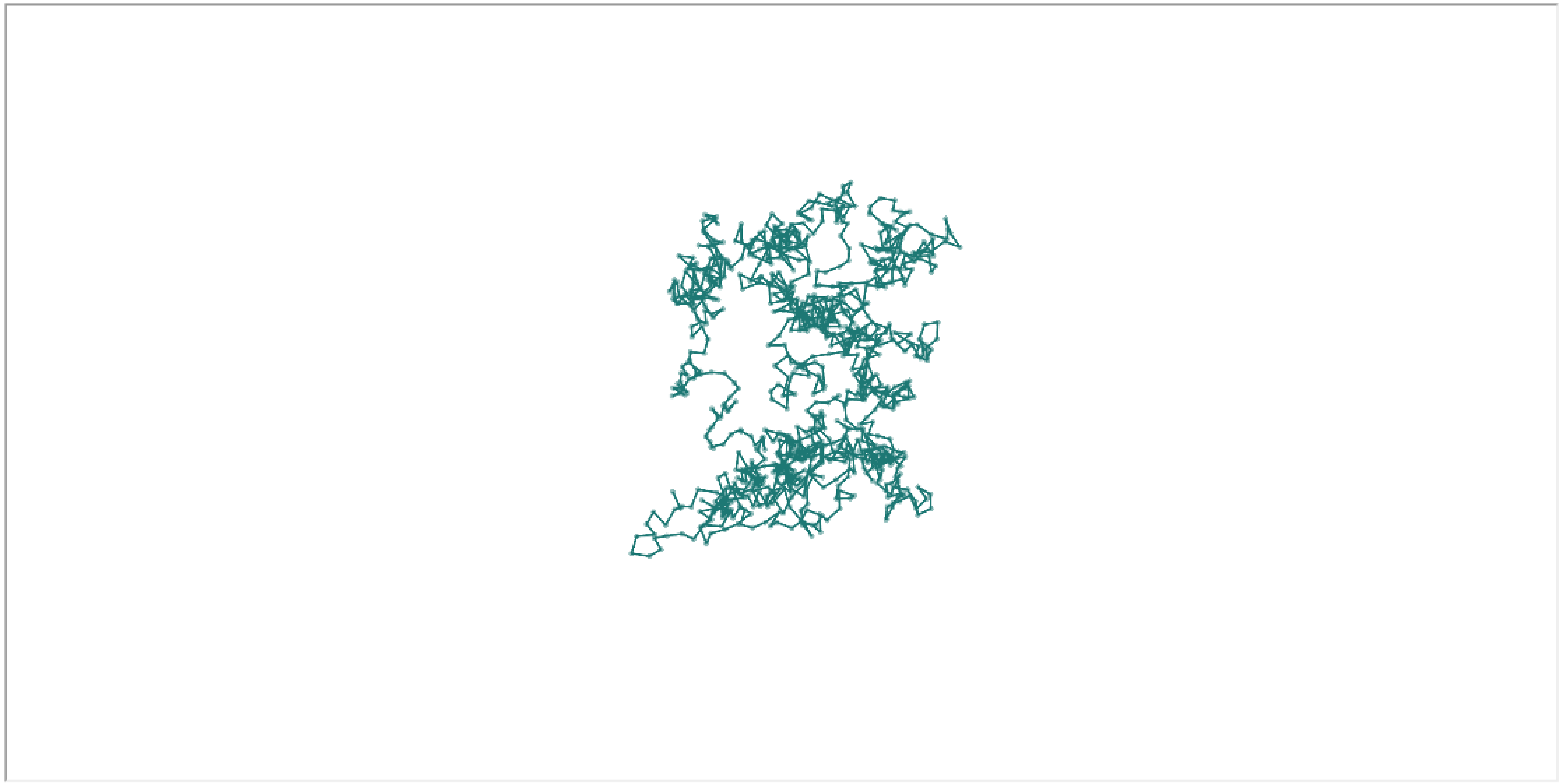


Fast intersection checks via bounding sphere refinements



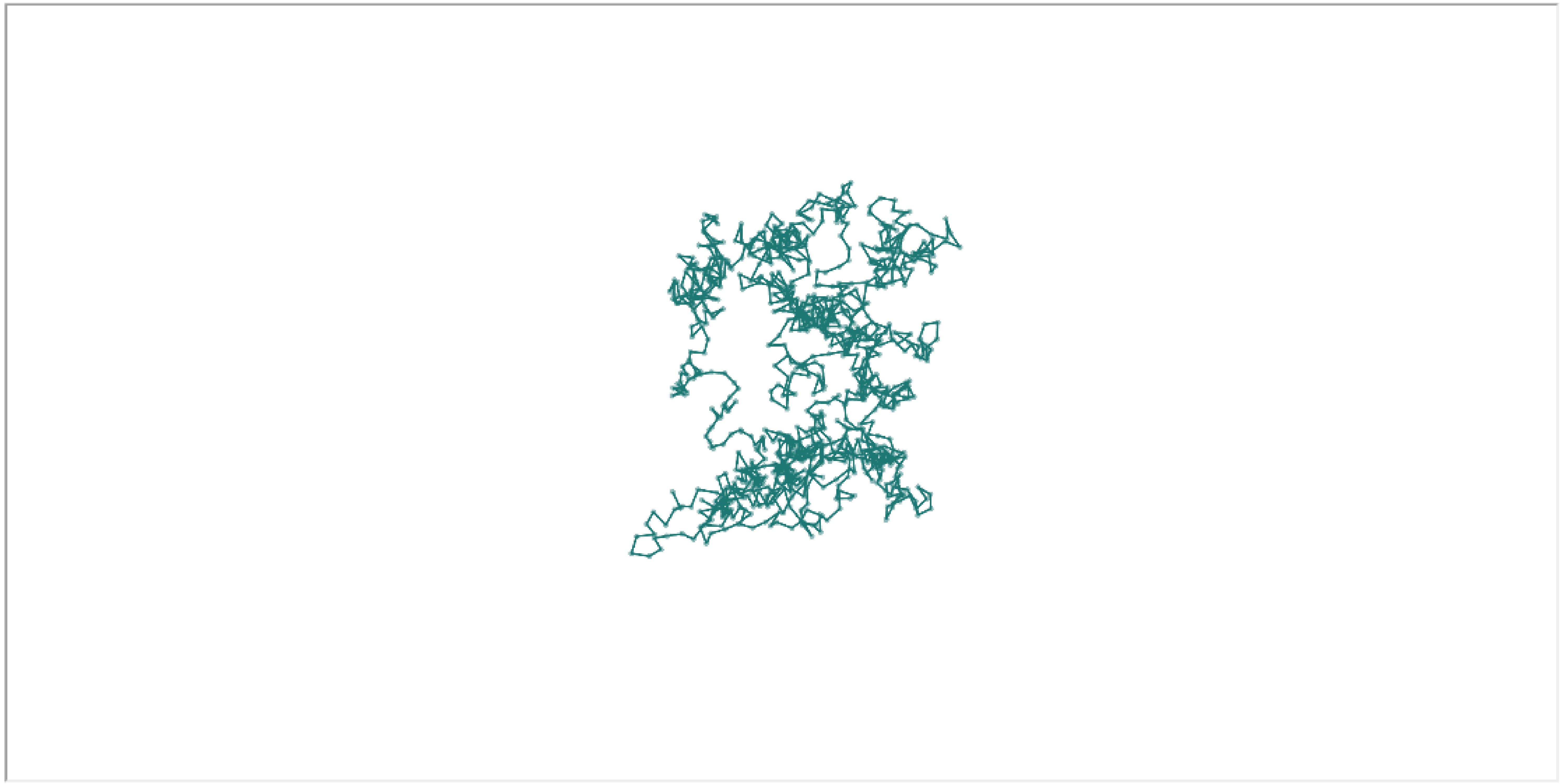
- Check segment-quadrilateral intersection via Möller-Trumbore

Does it work? Is topology conserved?



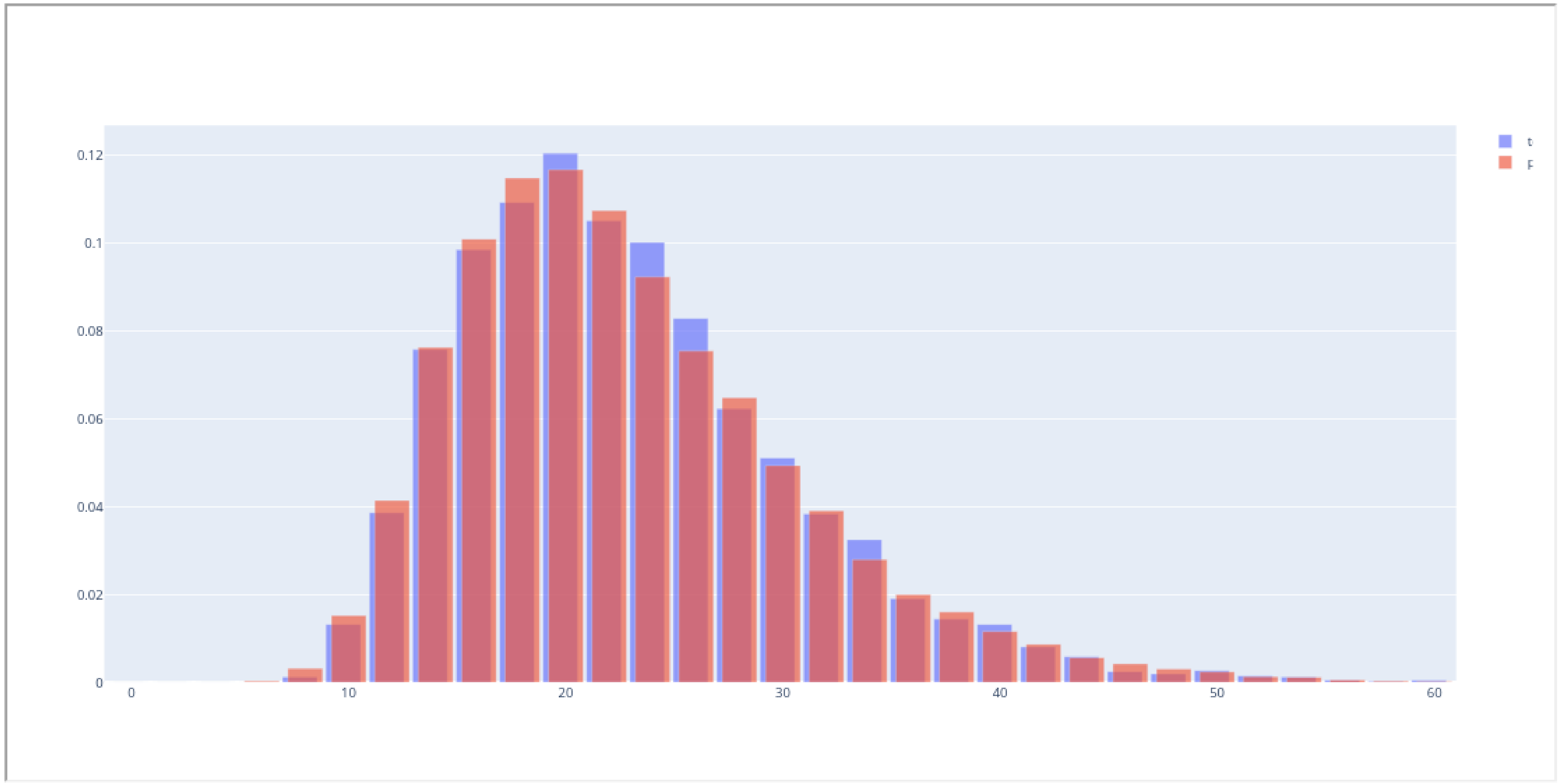
- 1024 edge square after $\approx$ 250k pivots
- Still an unknot

Does it work? Is topology conserved?



- 1024 edge square after $\approx$ 250k pivots
- Still an unknot
- Important aside — the topoly library is extremely helpful!

Does it work? Compare R_g histograms



- Generate 2^{12} length 256 unknots with the topoly library
- Generate 2^{14} length 256 unknots by pivots
- Close agreement

Does it work? Is it fast? Autocorrelation is everything

Warning: research still in progress

- Had great difficulty computing reliable autocorrelation time estimates

Does it work? Is it fast? Autocorrelation is everything

Warning: research still in progress

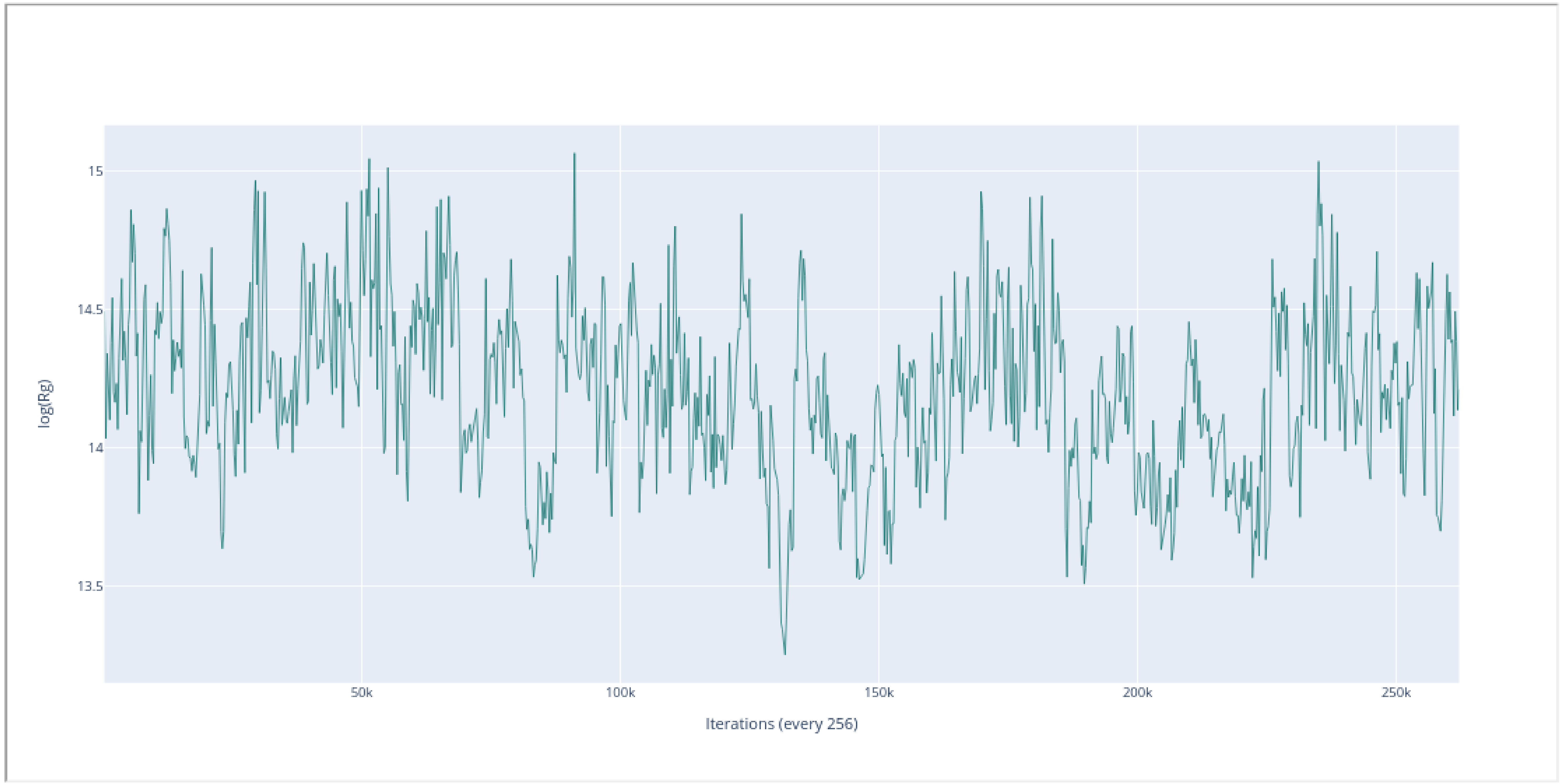
- Had great difficulty computing reliable autocorrelation time estimates
  - Windowing method via EMCEE python module
  - Log-binning method Wallerberger (2018)

Does it work? Is it fast? Autocorrelation is everything

Warning: research still in progress

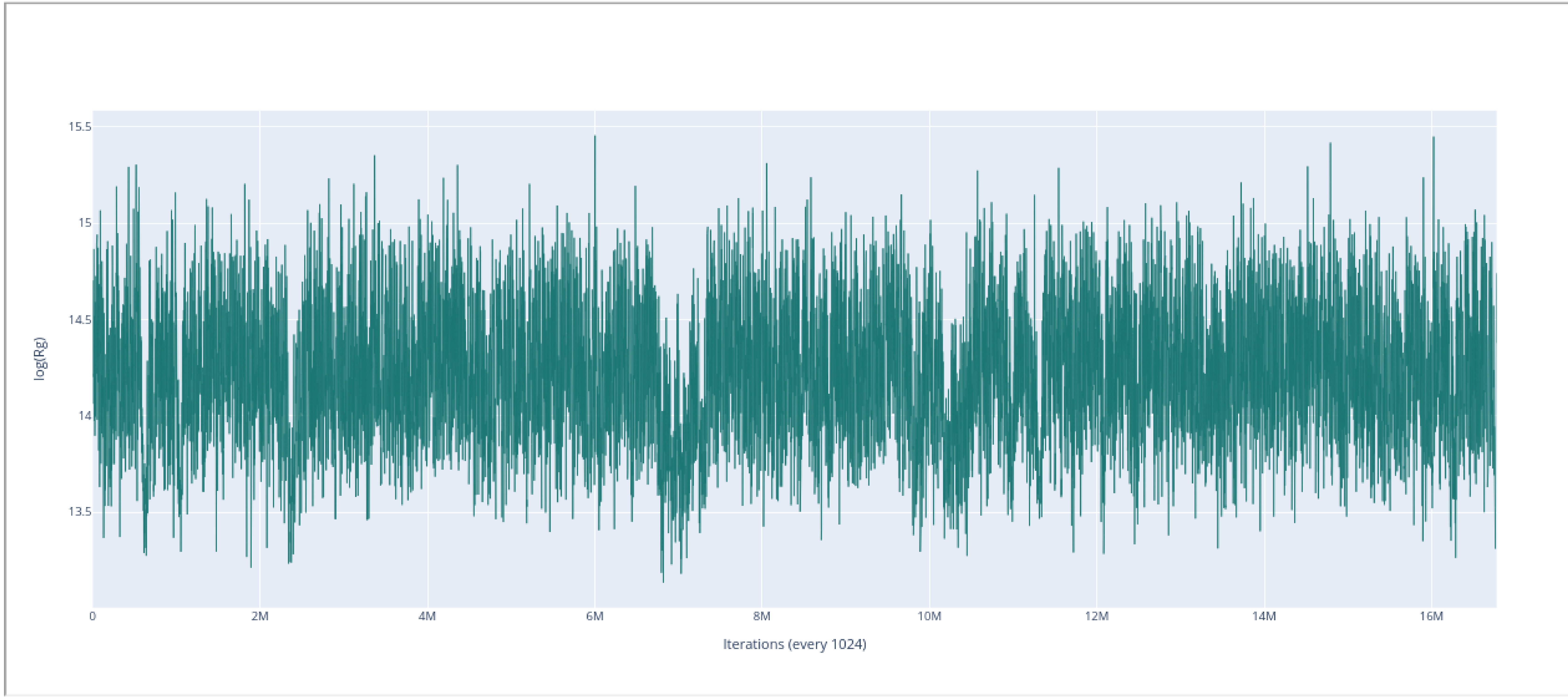
- Had great difficulty computing reliable autocorrelation time estimates
  - Windowing method via EMCEE python module
  - Log-binning method Wallerberger (2018)
- Huh? What is going on

Plot evolution of R_g with iterations



- Unknot length 256, every 256th iteration shown
- Looks okay, but those *"canyons"* are worrying

Plot evolution of R_g with iterations



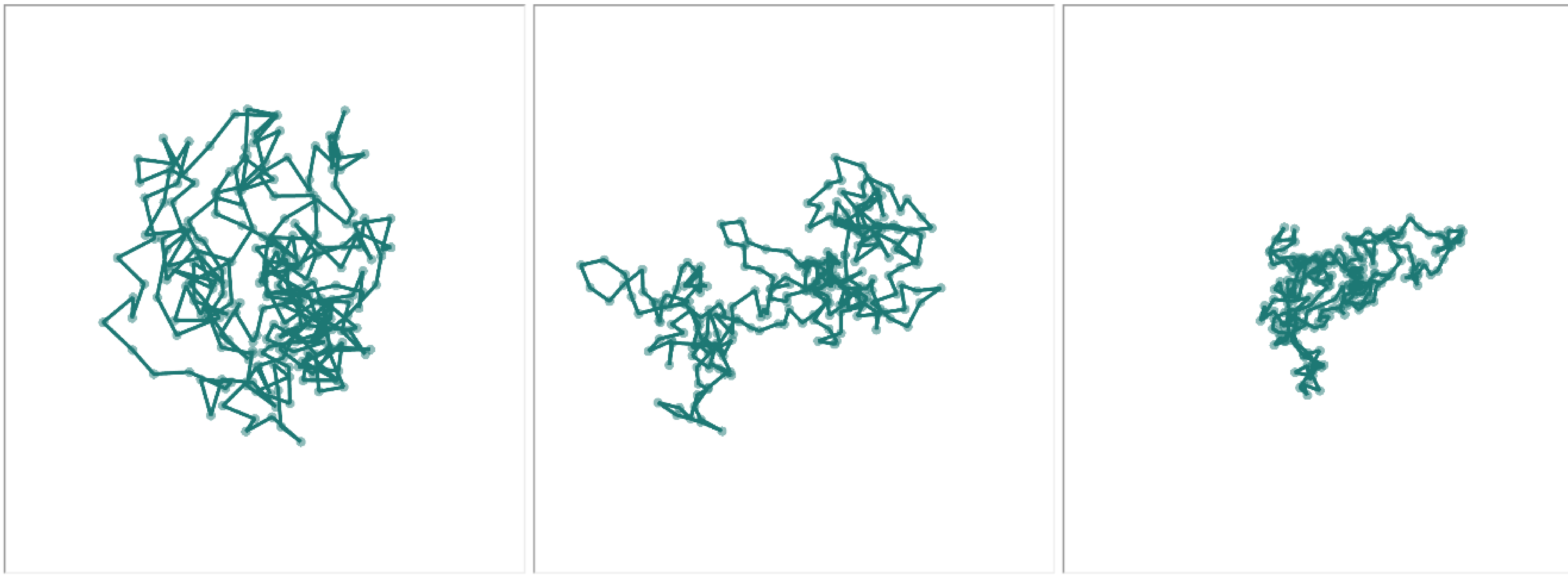
- Unknot length 256, every 1024th iteration shown
- Now *"canyons"* are very worrying

Possibility 1

bugs in my code

Possibility 2

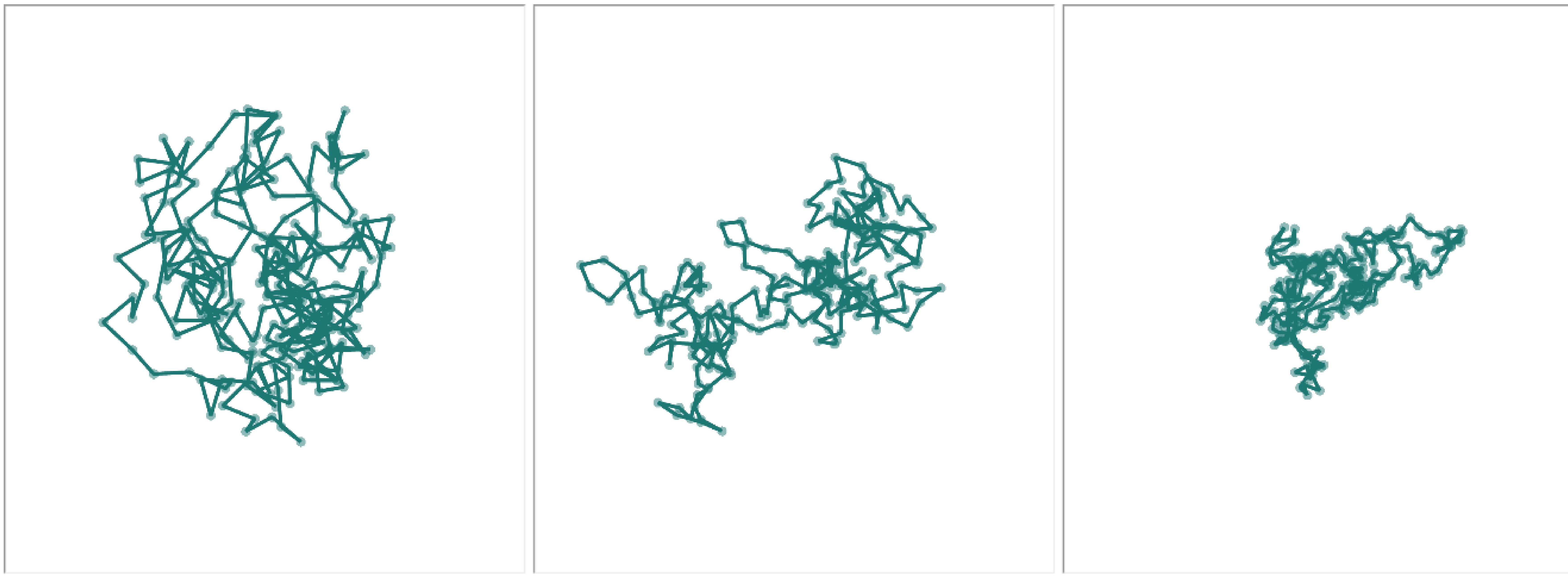
Compact conformations are not so rare



- Hard to pivot away from compact conformations

Possibility 2

Compact conformations are not so rare



- Hard to pivot away from compact conformations

Does not exclude Possibility 1

Is topological swelling to blame?

Is topological swelling to blame?

- Metric scaling of ERP and ERUnkots are different
  - ERP are compact — random walk universality class $\nu = \frac{1}{2}$
  - Believe that unknotted ERP swell — self-avoiding walk universality class $\nu \approx 0.6$

Is topological swelling to blame?

- Metric scaling of ERP and ERUnkots are different
  - ERP are compact — random walk universality class $\nu = \frac{1}{2}$
  - Believe that unknotted ERP swell — self-avoiding walk universality class $\nu \approx 0.6$
- By contrast — Metric scaling of SAP and SAUnkots are same
  - All SAP and unknotted SAPS have $\nu \approx 0.6$

Is topological swelling to blame?

- Metric scaling of ERP and ERUnkots are different
  - ERP are compact — random walk universality class $\nu = \frac{1}{2}$
  - Believe that unknotted ERP swell — self-avoiding walk universality class $\nu \approx 0.6$
- By contrast — Metric scaling of SAP and SAUnkots are same
  - All SAP and unknotted SAPS have $\nu \approx 0.6$
- Conformations in the ambient ERP space are typically more compact
- Does topology preservation restrict diffusion-via-pivots to a quasi-ergodic subspace?

Is topological swelling to blame?

- Metric scaling of ERP and ERUnkots are different
  - ERP are compact — random walk universality class $\nu = \frac{1}{2}$
  - Believe that unknotted ERP swell — self-avoiding walk universality class $\nu \approx 0.6$
- By contrast — Metric scaling of SAP and SAUnkots are same
  - All SAP and unknotted SAPS have $\nu \approx 0.6$
- Conformations in the ambient ERP space are typically more compact
- Does topology preservation restrict diffusion-via-pivots to a quasi-ergodic subspace?
- Are these just from bugs in my code?

What now?

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress
- Nathan and/or Nick code up algorithm independently

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress
- Nathan and/or Nick code up algorithm independently
- Alternate / better encoding of polygon (à la Cantarella?)

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress
- Nathan and/or Nick code up algorithm independently
- Alternate / better encoding of polygon (à la Cantarella?)
- Consider mixing BFACF moves with pivots
    - triangle $\leftrightarrow$ quadrilateral moves
    - much fun with data structures and lazy evaluation

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress
- Nathan and/or Nick code up algorithm independently
- Alternate / better encoding of polygon (à la Cantarella?)
- Consider mixing BFACF moves with pivots
  - triangle $\leftrightarrow$ quadrilateral moves
  - much fun with data structures and lazy evaluation
- Volume exclusion version of algorithm, on or off lattice
  - picking valid pairs of vertices on lattice is not $O(1)$

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress
- Nathan and/or Nick code up algorithm independently
- Alternate / better encoding of polygon (à la Cantarella?)
- Consider mixing BFACF moves with pivots
  - triangle $\leftrightarrow$ quadrilateral moves
  - much fun with data structures and lazy evaluation
- Volume exclusion version of algorithm, on or off lattice
  - picking valid pairs of vertices on lattice is not $O(1)$
- Allow reversals of segments — requires very careful tree & data-structure hackery

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress
- Nathan and/or Nick code up algorithm independently
- Alternate / better encoding of polygon (à la Cantarella?)
- Consider mixing BFACF moves with pivots
    - triangle $\leftrightarrow$ quadrilateral moves
    - much fun with data structures and lazy evaluation
- Volume exclusion version of algorithm, on or off lattice
    - picking valid pairs of vertices on lattice is not $O(1)$
- Allow reversals of segments — requires very careful tree & data-structure hackery
- Work continues (slowly)

What now?

- Much debugging and ~~swearing~~ more debugging — in (punctuated) progress
- Nathan and/or Nick code up algorithm independently
- Alternate / better encoding of polygon (à la Cantarella?)
- Consider mixing BFACF moves with pivots
    - triangle $\leftrightarrow$ quadrilateral moves
    - much fun with data structures and lazy evaluation
- Volume exclusion version of algorithm, on or off lattice
    - picking valid pairs of vertices on lattice is not $O(1)$
- Allow reversals of segments — requires very careful tree & data-structure hackery
- Work continues (slowly)

Many thanks to the organisers for today

Richard

Richard

- Took a course in asymptotics with him around 1994(?)
- Coauthored 10 papers with him, including my first paper in 1996/7
- I learned much about bijections, constant terms, generating functions, graphic-design, …
- Also much about teaching proof & logic

Richard

- Took a course in asymptotics with him around 1994(?)
- Coauthored 10 papers with him, including my first paper in 1996/7
- I learned much about bijections, constant terms, generating functions, graphic-design, …
- Also much about teaching proof & logic
- We always enjoyed a good grumble

YXE after cancelled flight June 2015

Richard

- Took a course in asymptotics with him around 1994(?)
- Coauthored 10 papers with him, including my first paper in 1996/7
- I learned much about bijections, constant terms, generating functions, graphic-design, …
- Also much about teaching proof & logic
- We always enjoyed a good grumble
- Drank many coffees (and some beers)

Richard

- Took a course in asymptotics with him around 1994(?)
- Coauthored 10 papers with him, including my first paper in 1996/7
- I learned much about bijections, constant terms, generating functions, graphic-design, …
- Also much about teaching proof & logic
- We always enjoyed a good grumble
- Drank many coffees (and some beers)
- He made a big impact on my mathematics; how I do it, how I present it, and how I teach it