FOUNDATIONS · READ MAPPING

How minimap2 maps a long read

The **seed → chain → align** idea behind fast, accurate long-read alignment — and **minisplice**, the deep-learning step that sharpens splice junctions.

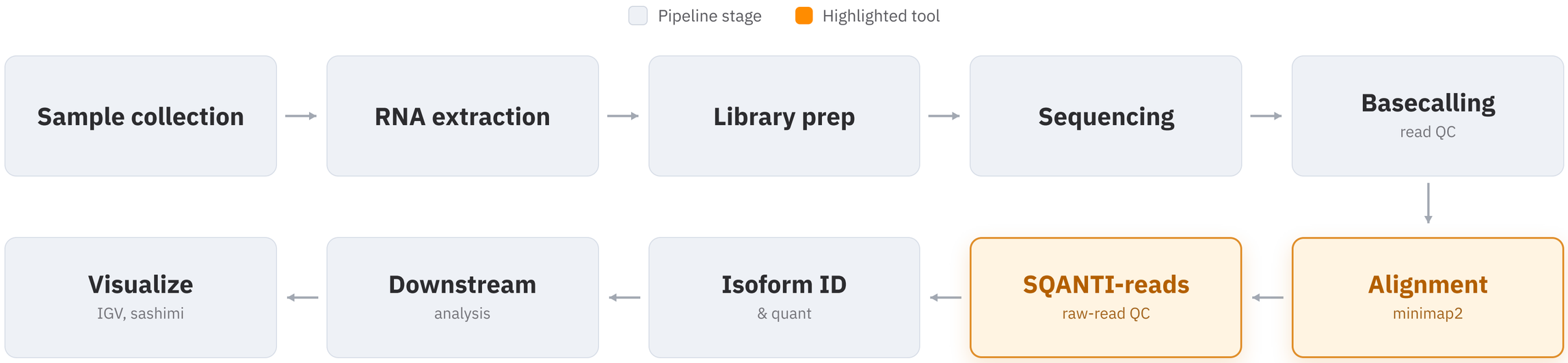
1 Seed | 2 Chain | 3 Align | + minisplice

minimap2 · Li, *Bioinformatics* 2018
minisplice · Yang, Huang & Li, *Algorithms Mol. Biol.* 2025

Funded by the European Union

From sample to isoforms

Every long-read transcriptomics experiment runs this pipeline. This course zooms in on the two highlighted tools — alignment with **minimap2** and raw-read QC with **SQANTI-reads**.



THE PIPELINE AT A GLANCE

Index once, then seed → chain → align

One minimizer index of the reference, then three steps run on **every read**.

THE CORE TRICK

Minimizers

Within each sliding window minimap2 keeps just the **smallest k-mer** — a minimizer. Two sequences that share DNA pick the **same** minimizers, so they match without comparing every base.

a sliding window of consecutive k-mers keeps one minimizer

CGT **ACGTG** CAT **GGTCA** TG **CAATG** CCA **TGACC** TAG

only the orange k-mers are indexed — far fewer seeds, the same matches found

0 PREP — Build index

Scan the reference once; store every minimizer and its position in a hash table.

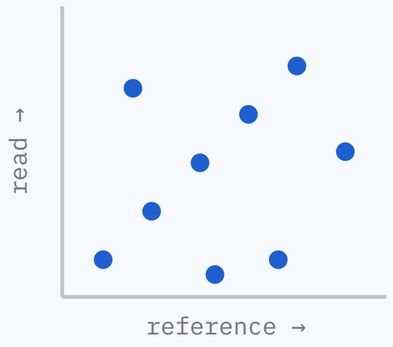
reference genome

MINIMIZER	POSITION
ACGTG	chr1:128
GGTCA	chr1:256
CAATG	chr1:392

once per reference

1 STEP 1 — Seed

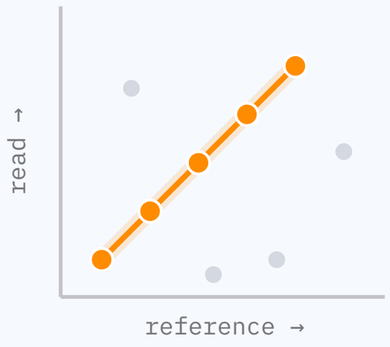
Look up the read's minimizers. Each hit is an **anchor** — a seed shared by read and reference.



minimizer hits → anchors

2 STEP 2 — Chain

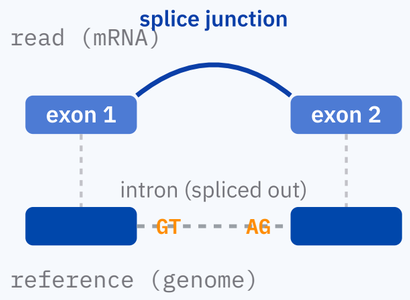
Dynamic programming links **collinear** anchors into the best-scoring chain — where the read fits.



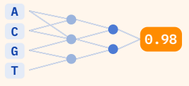
collinear anchors → chain

3 STEP 3 — Align

Base-level DP fills the gaps between anchors for an exact alignment; long gaps become **introns**.



→ CIGAR in BAM / PAF



minisplice — sharper splice junctions

A small **1D-CNN** scores every **GT**/**AG** splice site in the genome; minimap2 (r1285+) uses the scores to place introns more accurately — especially on noisy long RNA-seq reads.

NEW · TRANSCRIPTOMICS

~7k parameters

THE CORE TRICK · FROM LETTERS TO NUMBERS

Computers compute on numbers, not letters

DNA is written in A, C, G, T — but a computer can only store, compare and sort **numbers**. So each k-mer must become one. Three families do it differently.

Step one is shared by all three — **map each base to a digit:**

A	C	G	T
0	1	2	3

the difference is how those digits become **one number** for the whole k-mer

Encoding

keep the exact value · 2-bit

A 00 C 01 G 10 T 11

ACGTT → 00 01 10 11 **11**

ACGTG → 00 01 10 11 **10**

the number *is* the sequence — reversible
similar in → similar out

✕ grows with k — no small fingerprint

Hashing MINIMAP2'S PICK

scramble to a small fingerprint

same k-mer → same tag

ACGTT → **8821**

ACGTG → **0342**

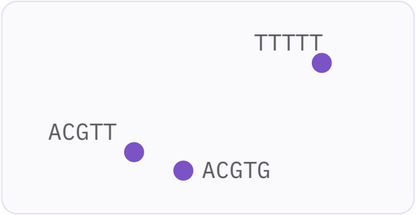
one base differs → unrelated value
similar in → unrelated out

✓ tiny fingerprint — fast & light

✓ exact identity + fair order

Embedding

place it by meaning · ML



nearby = similar
distance carries meaning

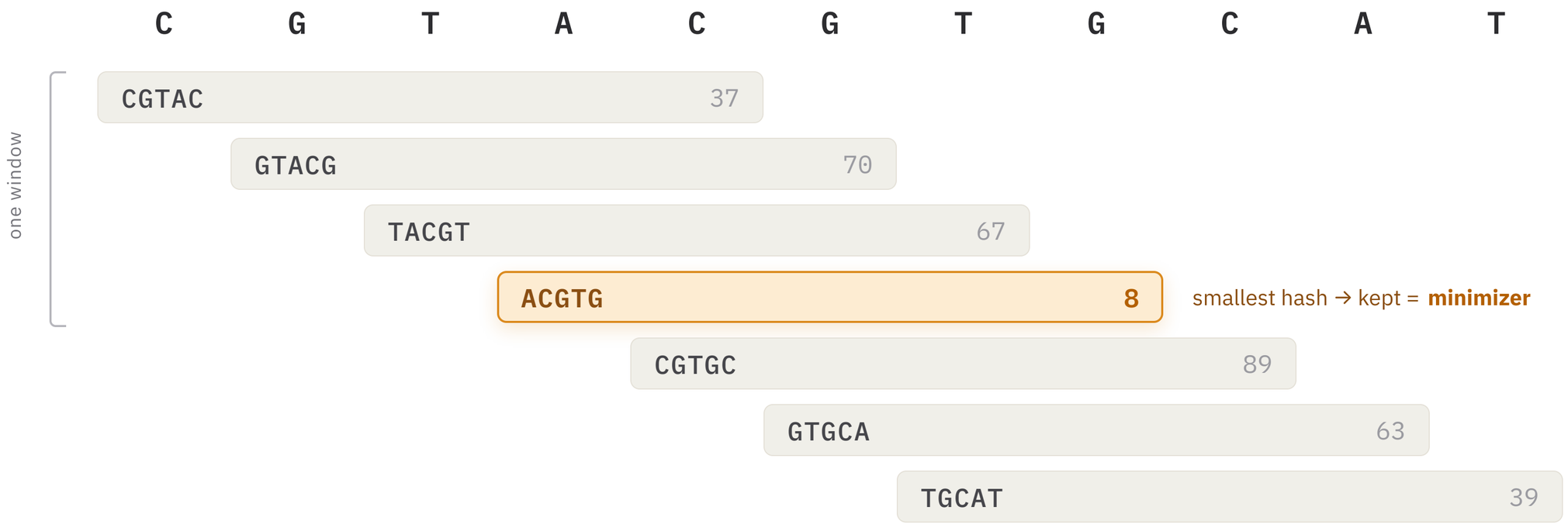
✕ slow & memory-hungry (ML vectors)

✕ fuzzy match, not exact identity

Encoding keeps a **bulky exact value**; embedding needs **slow, memory-hungry vectors**. Only **hashing** shrinks each k-mer to one **tiny fingerprint** that is also an exact, fairly-ordered key — that's why minimap2 hashes.

In each window, keep the smallest hash

A k-mer starts at every base, so they overlap — the smallest-hash one is the **minimizer**.



Grey k-mers overlap the winner — **skipped, not removed**. Matching DNA always picks the same minimizer → a shared **seed**.

STEP 0 · BUILD THE INDEX

Index the reference once, up front

- Built **once, up front** — scan the whole reference and keep only its minimizers.
- Store each in a **hash table: minimizer → position(s)** in the genome.
- A minimizer that recurs just **collects all its positions in one row** — like GGTCA at 256 and 1200.

THE INDEX IS A HASH MAP

`minimizer → { pos, pos, … }`

Built once, reused for every read — one lookup returns every candidate location.

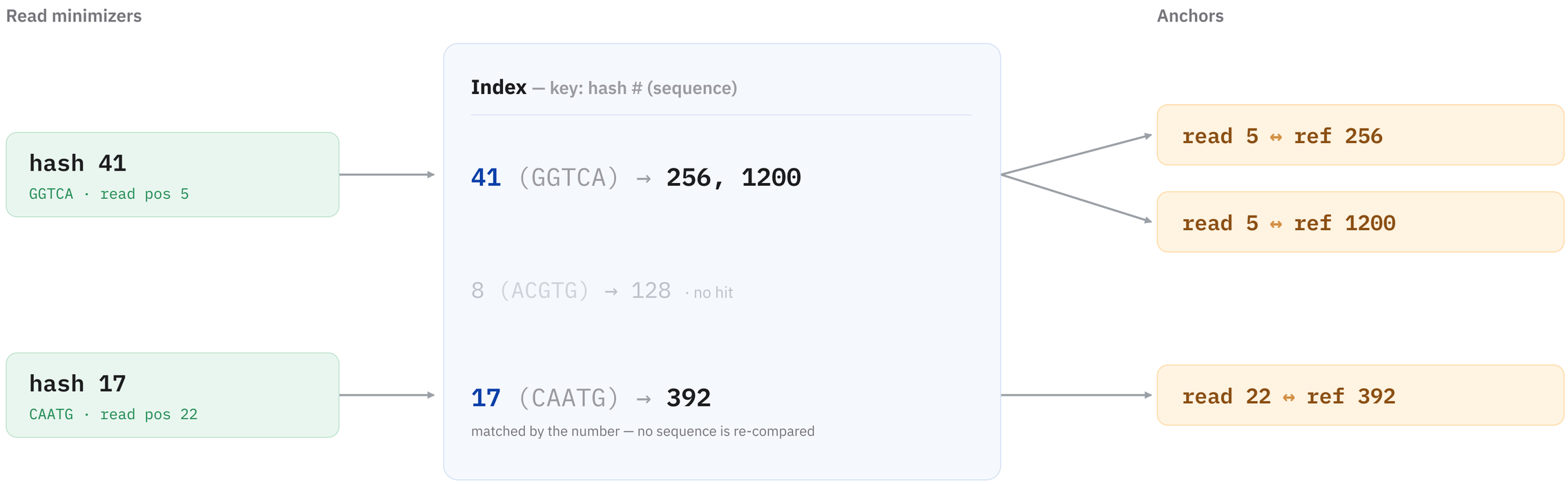
scan once — record each minimizer and where it sits

MINIMIZER → POSITIONS	
ACGTG → 128	
GGTCA → 256, 1200	ONE ROW · TWO POSITIONS
CAATG → 392	

a repeated minimizer collects all its positions in one row

STEP 1 · SEED

Matching minimizers become anchors



The match is **number = number** — the read's **41** finds key **41**, and each key also stores its sequence (**41 = GGTCA**) so you can see what the number stands for. **GGTCA hits two places → two anchors**; a key the read doesn't carry (8) is simply a miss. Highly repetitive minimizers are skipped to stay fast.

AN ANCHOR IS A TRIPLE

`(x, y, w)`

ref position x · read position y · match length w

STEP 1 · SEED

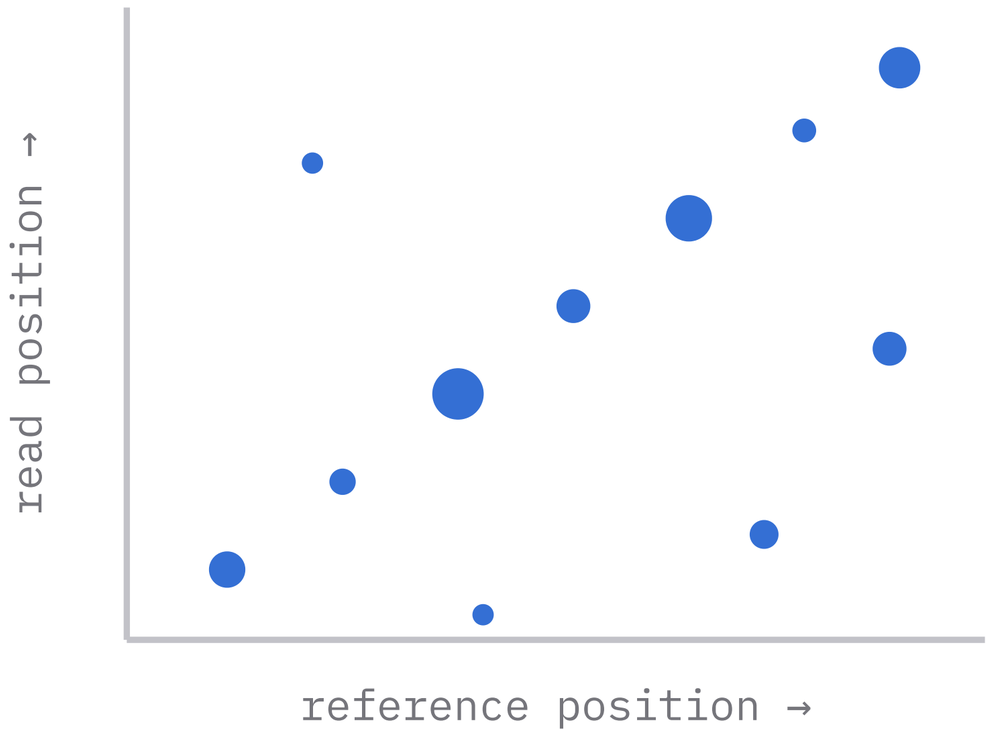
Each anchor is a dot, sized by its match length

- Every anchor is one point at **(reference position, read position)** — its coordinates (x, y).
- The dot's **size is the match length w** — a longer exact match draws a bigger dot.
- The true alignment's anchors line up on a **diagonal**; repeats and noise scatter off it.

THE DOT IS THE WHOLE ANCHOR

`(x, y, w)`

position **(x, y)** places the dot · length **w** sets its size.



short w long w · dot size = anchor length

true anchors line up on a diagonal — what chaining threads next

STEP 2 · CHAIN

Chaining connects anchors that line up

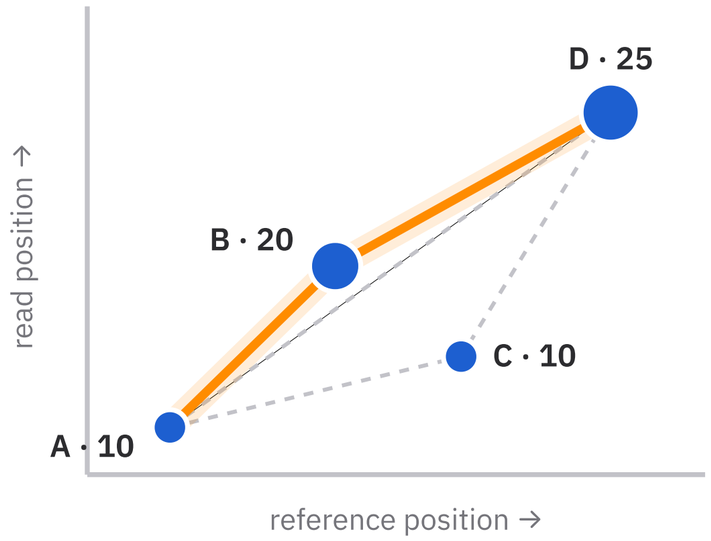
- **Collinear** anchors — same order and spacing on read and reference — likely belong to one true alignment.
- DP scores every route into an anchor: **reward** the matched bases it adds (α), **subtract** a concave gap cost (β) for the jump.
- Keep only the best predecessor per anchor; a look-back of ≈ 50 stays near-linear. **Chaining alone** already maps accurately.

$$f(i) = \max\{ \max_j [f(j) + \alpha(j,i) - \beta(j,i)], w_i \}$$

$f(j)$ best score of a chain ending at an earlier anchor j

$\alpha(j,i)$ reward · matched bases i adds · $\beta(j,i)$ gap cost for the jump

w_i anchor i's own weight — the option to start a fresh chain



Best route ending at D?

A→D : 10 + 10 − 5 = 15

A→B→D : 20 + 10 − 5 = 25 ✓

A→C→D : 10 + 10 − 5 = 15

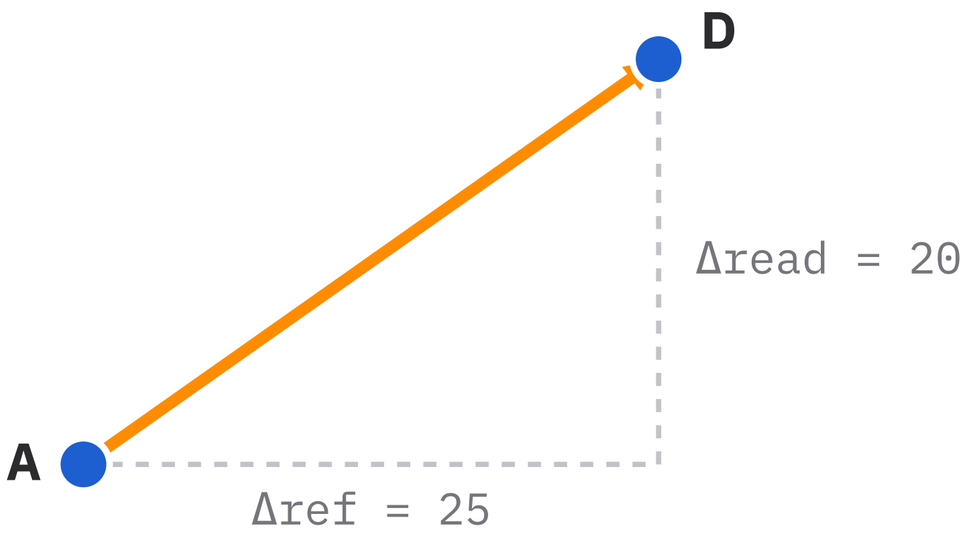
best route → **A→B→D** (25)

dot size = anchor weight w · candidates tie on the gap (−5), so the best sub-route wins: A→B→D

STEP 2 · CHAIN · SCORING ONE LINK

Where each number comes from

ref moves 25, read moves 20 → off by 5



A→D : 10 + 10 − 5 = 15

A's score · D's bases · gap

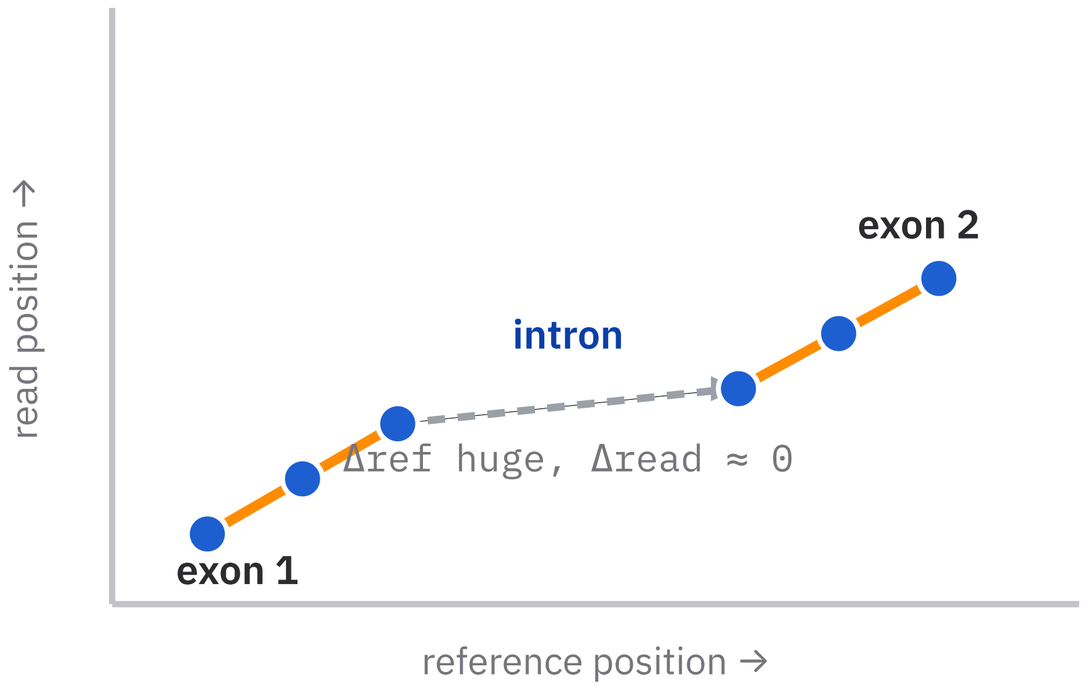
Where each number comes from

- **10** = score of the route up to A — that's **f(j)** (A starts the chain)
- **10** = D's own matching bases — its weight, the reward **α(j,i)**
- **5** = gap = |Δref − Δread| = |25 − 20| — the cost **β(j,i)**

FOR RNA · SPLICED ALIGNMENT

Long gaps become introns

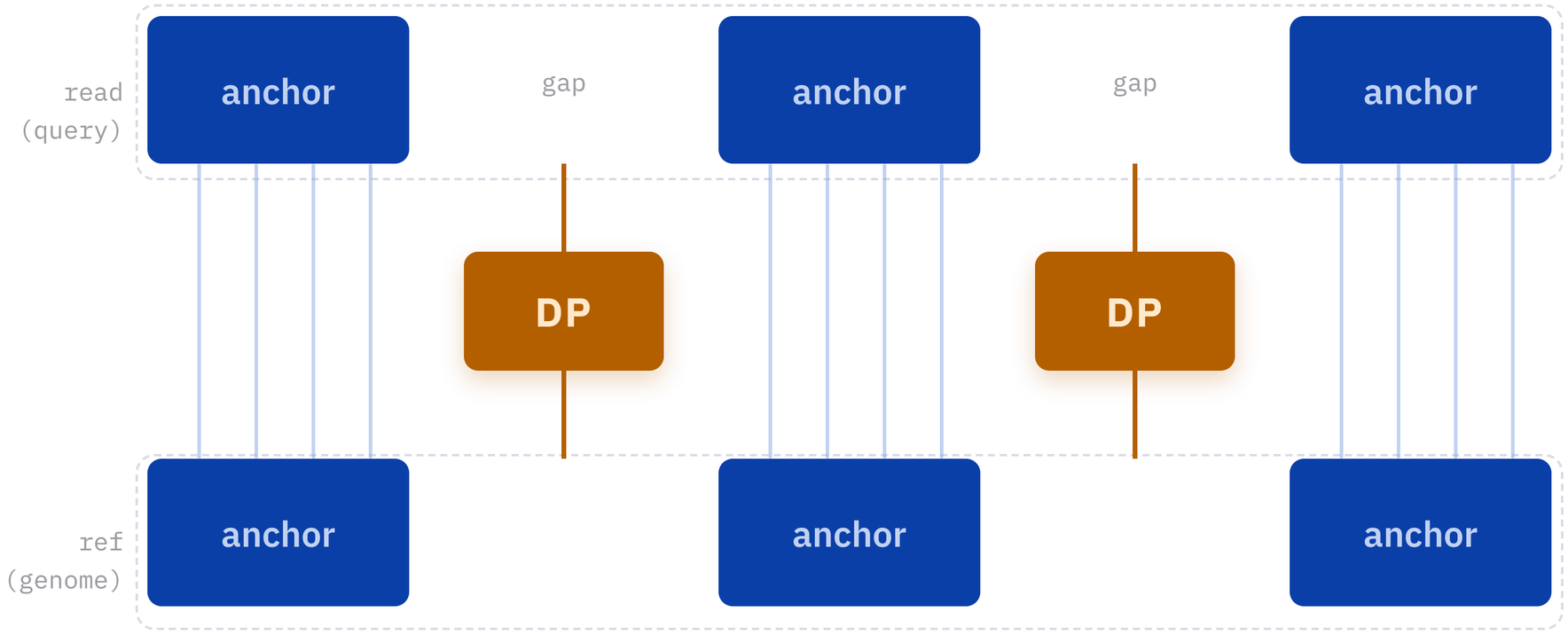
- An mRNA read **skips introns**. A normal gap cost would punish that huge reference jump — unfair to RNA.
- Splice mode **expects introns**, so the long reference gap is **cheap**: the chain crosses it in a single step.
- minimap2 checks for canonical **GT**...**AG** splice signals at the ends to place the junction exactly.



normal scoring punishes the big reference gap · splice mode makes it cheap, so the chain crosses the intron — pinned by GT…AG

STEP 3 · ALIGN

Anchors are the skeleton — DP fills the gaps



anchor = exact match DP = fills exact edits

each anchor is an exact-match block — base-level DP runs only in the short gaps between them

STEP 3 · ALIGN

Inside the DP: an edit-distance grid

reference →

read ↓

	–	G	A	T	C
–	0	1	2	3	4
G	1	↘ 0	1	2	3
A	2	1	↘ 0	→ 1	2
C	3	2	1	1	↘ 1

aligning read **GAC** to ref **GATC** — the cheapest path costs 1 edit

↘ align two bases (= / X)

→ gap in read (D)

↓ gap in ref (I)

cells on the best path

final cost = total edits

STEP 3 · ALIGN

The exact edits become a CIGAR

read	G	A	T	C	A	C	A	T	–	A
	=	=	=	X	=	=	I	=	D	=
ref	G	A	T	A	A	C	–	T	G	A

CIGAR 3= 1X 2= 1I 1= 1D 1=

by default minimap2 emits **6M1I1M1D1M** · run with **--eqx** for the = / X split shown

= match · X mismatch · I insertion (read) · D deletion (ref)

PAUSE & DISCUSS · CHECKPOINT

Quick questions — does the pipeline hold together?

1

At **seed**, do we match letters or numbers — and why does it matter?

2

What does **chaining** reward, and what does it penalise?

3

What is a **CIGAR**, and which step emits it?

CHALLENGE · THINK IT THROUGH HARD

The micro-exon problem

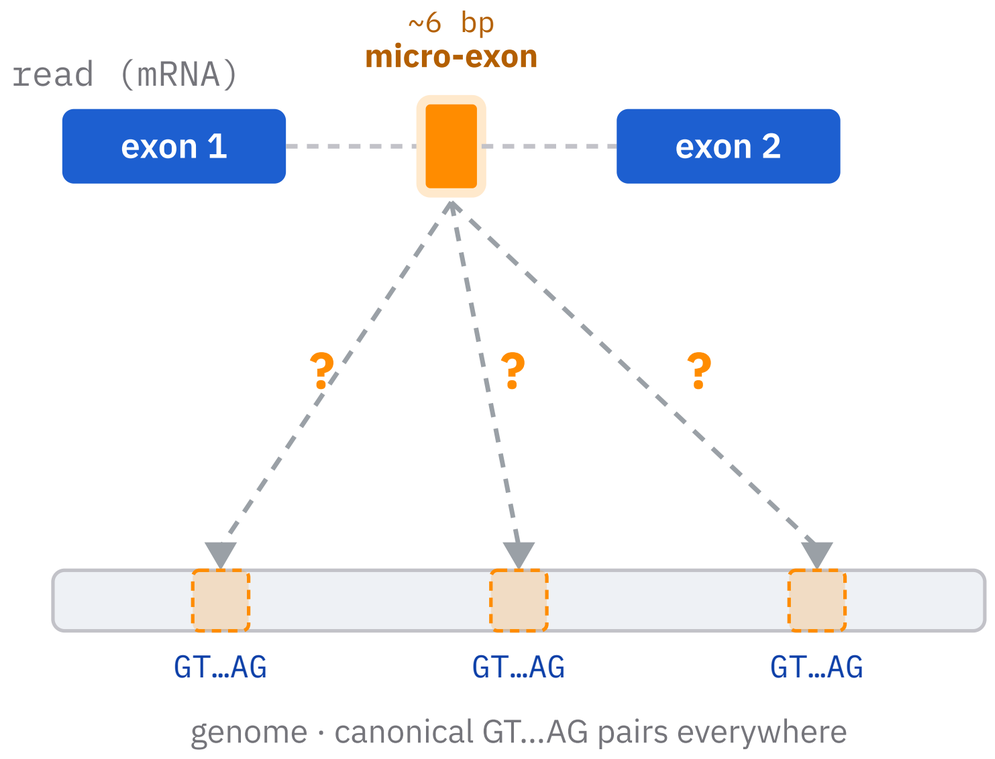
A **micro-exon** — sometimes just a handful of bases — sits between two long introns.

Its seed is so short it lands at **several positions** in the genome, and *every* candidate is flanked by a perfectly canonical GT…AG.

THE QUESTION

So which junction is the real one — and how could the aligner ever decide?

long enough to seed? | chain scores tie | GT…AG is everywhere



one 6 bp exon, several candidate landing sites — each with a canonical GT…AG. Which is real?

DEEP-LEARNING ADD-ON · FOR RNA

minisplice — learned splice scoring

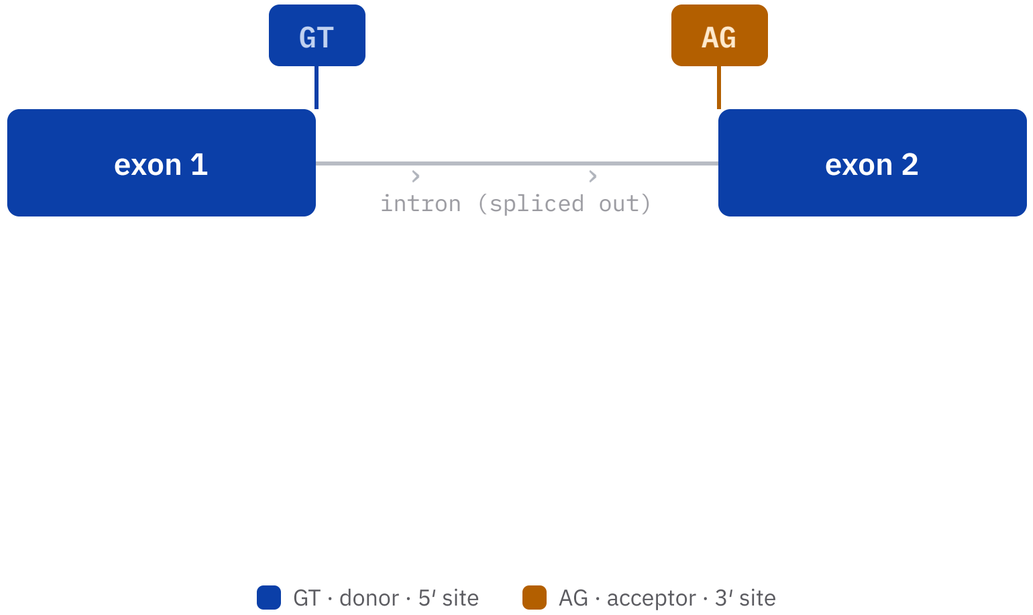
A small **1D-CNN** scores every **GT** donor and **AG** acceptor, so chaining drops each **intron** in exactly the right place — sharper junctions on noisy long RNA reads.

GTF labels → 202 bp windows → one-hot 4×202 → 1 : 3 pool → 1D-CNN → splice scores

Labels come from the genome + annotation

- The genome **FASTA** and the **GTF** annotation pin where every real splice junction sits.
- Donor sites (**5′**) always begin with **GT**; acceptor sites (**3′**) always end with **AG**.
- But **most** GT/AG dinucleotides aren't real sites — so minisplice learns to tell true junctions from look-alikes.

genome (FASTA) + annotation (GTF) → known junctions



Cut a 202 bp window around every GT and AG

GT **Donor arm** · 5′ splice site

202 bp window · centred on GT

exon ← −100 bp

+100 bp → intron

… G C T C A G C T G C A G **GT** A A G T G C C T G A C G …

AG **Acceptor arm** · 3′ splice site

202 bp window · centred on AG

intron ← −100 bp

+100 bp → exon

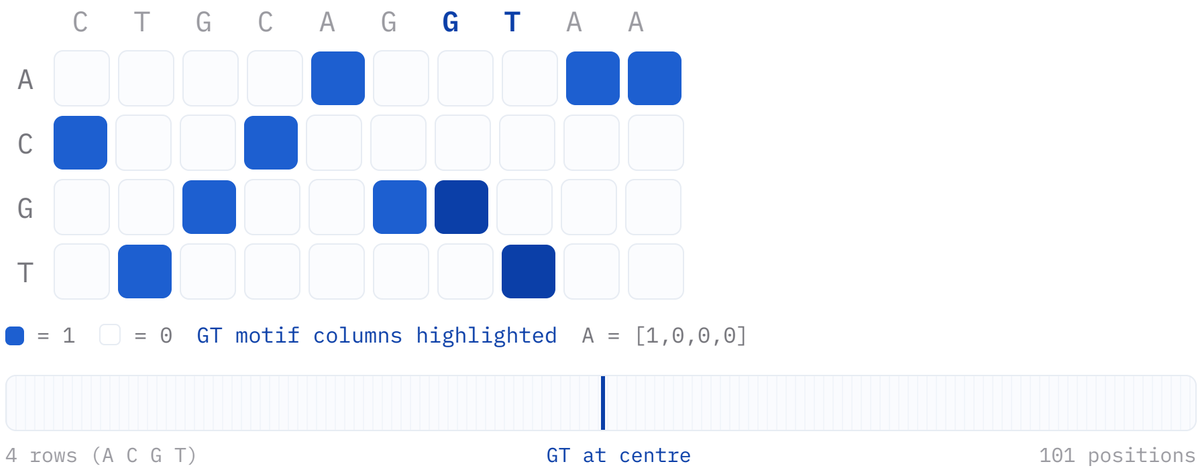
… T T C T T T C T C C C C **AG** G T G A A G C T G C A G …

the C/T-rich run just upstream is the polypyrimidine tract — a mark of a genuine 3′ site

same recipe for both arms · exactly 202 bp · the invariant dinucleotide pinned at the centre

One-hot encode, then balance the pool

a 10 bp slice across the donor boundary →



Donors and acceptors share **one pool**, then balanced against decoys:

GT AG | GT AG GT AG GT AG

2 real (1 donor + 1 acceptor) · 6 decoy GT/AGs **= 1 : 3**

decoys are GT/AG look-alikes sampled from non-junction positions — the model must separate true sites from them

A shared 1D-CNN scores each site

input 4 × 202
one-hot window

→

conv → 16 filters
scan for sequence motifs

→

global max-pool
keep the peak signal

→

dense → 16
compress to a score

→

0.97
splice probability

one shared network · ≈ 7,026 trainable parameters · used for both donor and acceptor windows

AT MAP TIME

minisplice scores every candidate GT/AG; a high score makes the intron gap **cheap**, so chaining lands each junction exactly — **≈ 94% of junctions exact** on ONT cDNA.

IN ONE BREATH

Seed, chain, align — fast because it skips the bulk

1 Seed

Minimizers turn shared DNA into a handful of **anchors**.

2 Chain

DP links the collinear anchors into the **best map**.

3 Align

Base-level DP fills the exact edits; long gaps are **introns**.

+ MINISPLICE Deep-learned splice scores place introns where splicing is genuinely likely — sharper junctions on noisy RNA.

≥ 30×
faster than other long-read mappers

~94%
exact splice junctions (ONT cDNA)

the default
long-read RNA aligner

Li, *Bioinformatics* 2018
Yang, Huang & Li, 2025