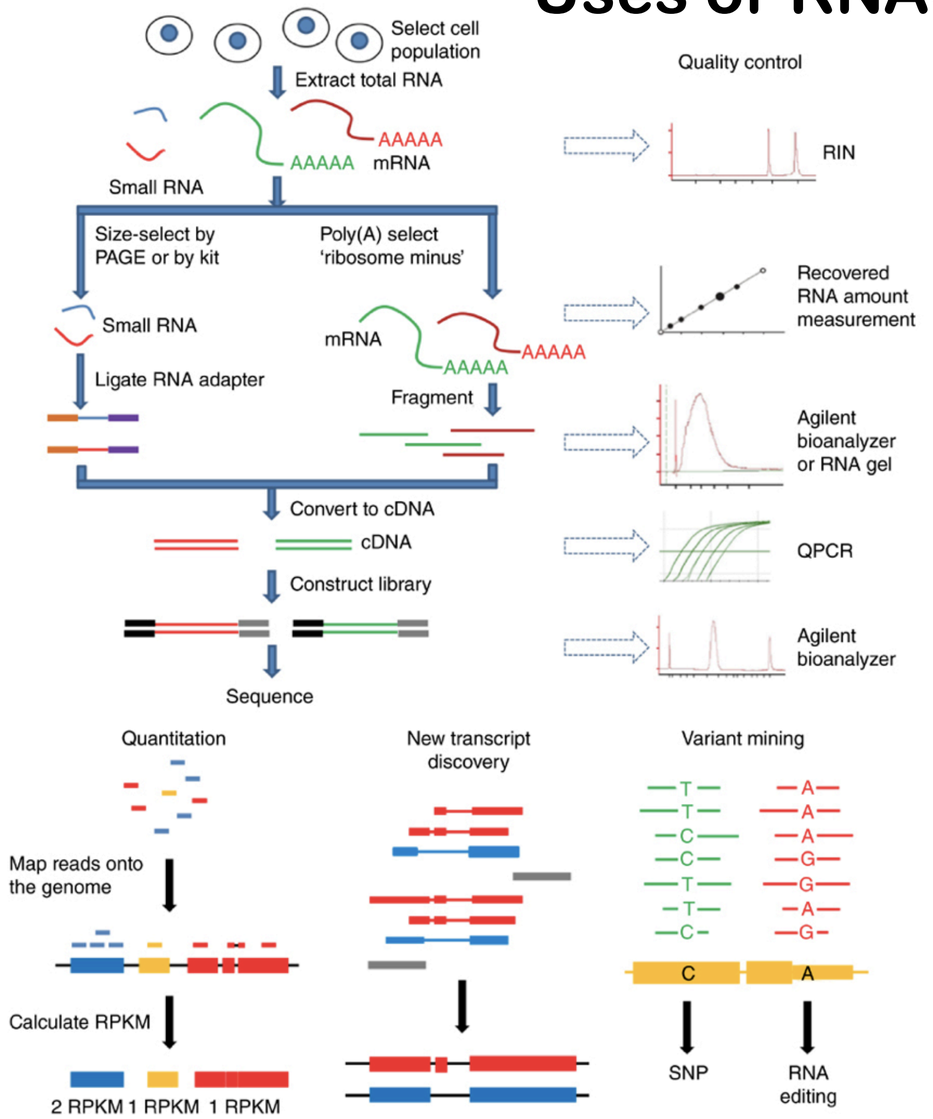
RNA-SEQ ALIGNMENT & TRANSCRIPT IDENTIFICATION

Uses of RNA-Seq are manifold

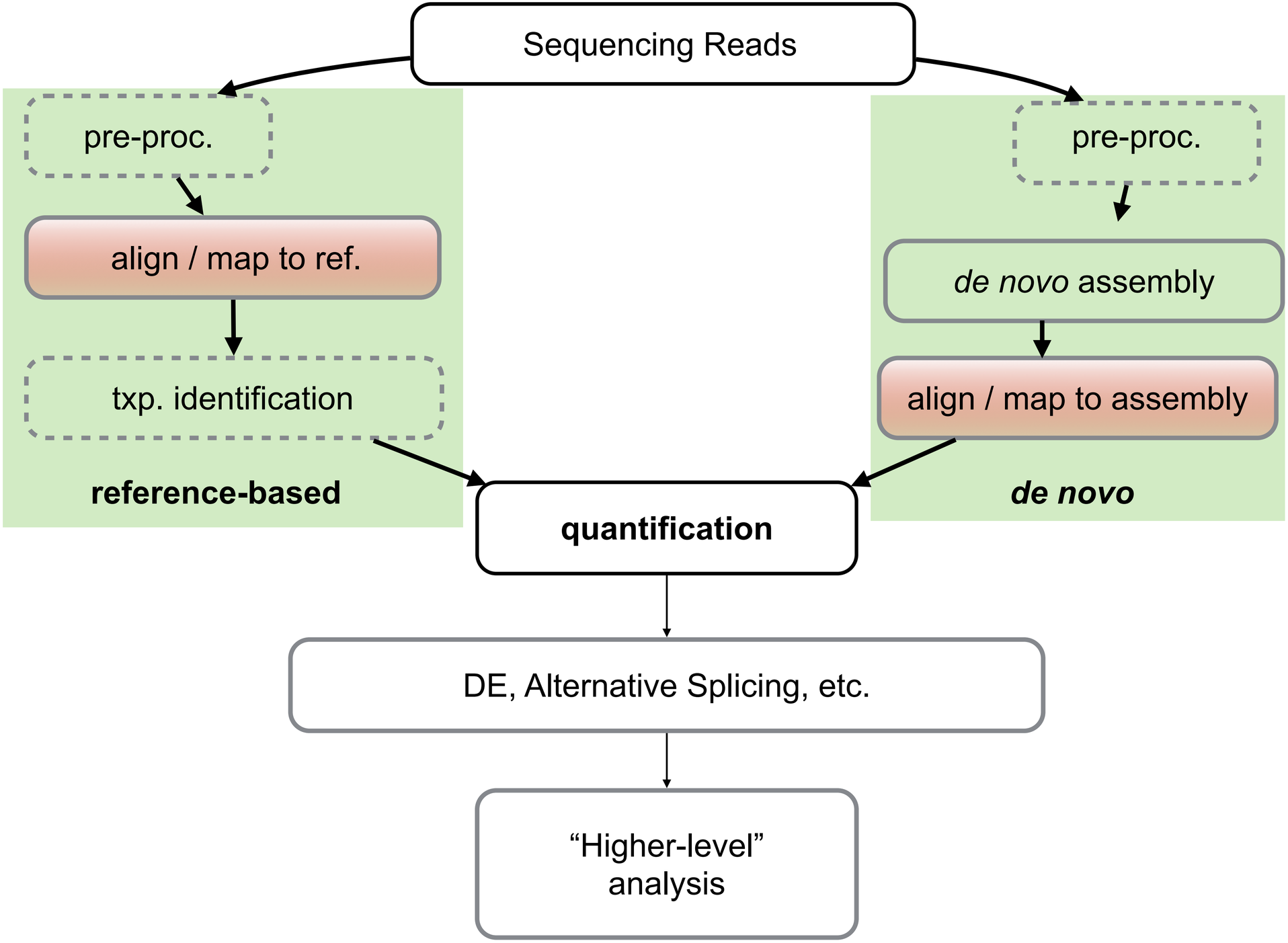
Zeng & Mortazavi, Nature Immunology 2012



Whole transcriptome analysis

- Quantification & differential expression
- Novel txp discovery
  - reference-based
  - *de novo*
- Variant detection
  - Genomic SNPs
  - RNA editing

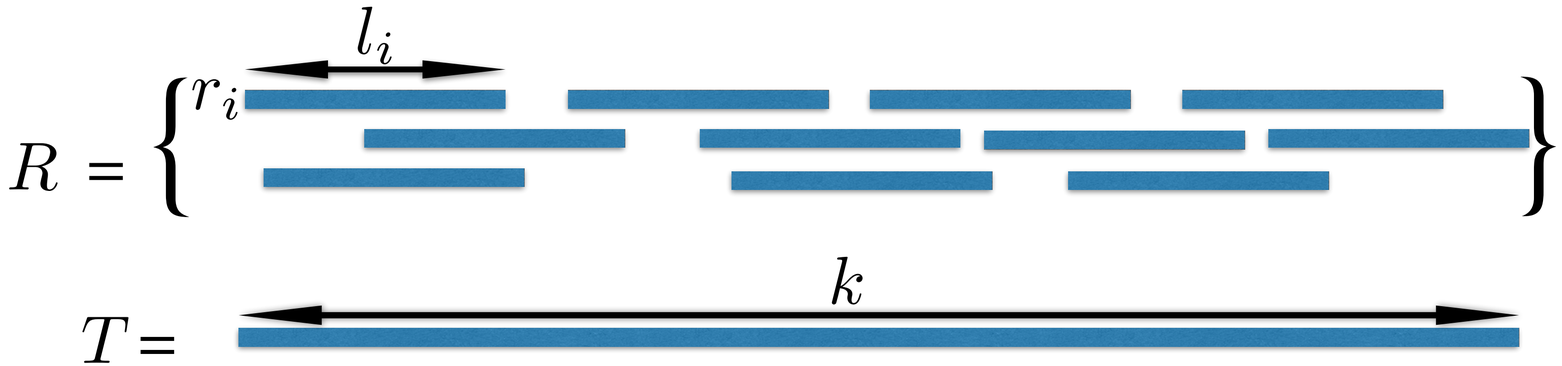
- What is dynamic & changing over time (as disease progresses)?
- What is tissue specific (in fetal development but not after)?
- What is condition specific (under stress conditions vs. not)?


Sequencing Reads
pre-proc.
align / map to ref.
txp. identification
reference-based
pre-proc.
de novo assembly
align / map to assembly
de novo
quantification
DE, Alternative Splicing, etc.
“Higher-level” analysis

What is the alignment problem?

Given: A collection of sequencing reads, and some target sequence (e.g. a genome)

Find: For each read, all locations where the read is within edit distance **ϵ** of the reference, and the edits that achieve this distance.



$$d : \Sigma^{|u|} \times \Sigma^{|v|} \to \mathbb{Z} \text{ and } \epsilon \text{ (maximum edit distance)}$$

Alignment Challenge

The best algorithms we have (and like the best that could exist) to compute the optimal alignment of two strings are *quadratic*

If we have N reads, each of length ℓ, and the genome is of length L, then applying the optimal algorithm at each possible position (to test the edit distance) is **O(N·ℓ·L)**

Consider a dataset with:

N = 20 x 10^6 reads

ℓ = 100

L = 3x10^9 nucleotides

and a processor that can do X = 3 x 10^9 operations / sec.

You'd wait about (N·ℓ·L) / X = 200,000,000 sec = 6.34 **years** to align your reads.

Alignment Challenge

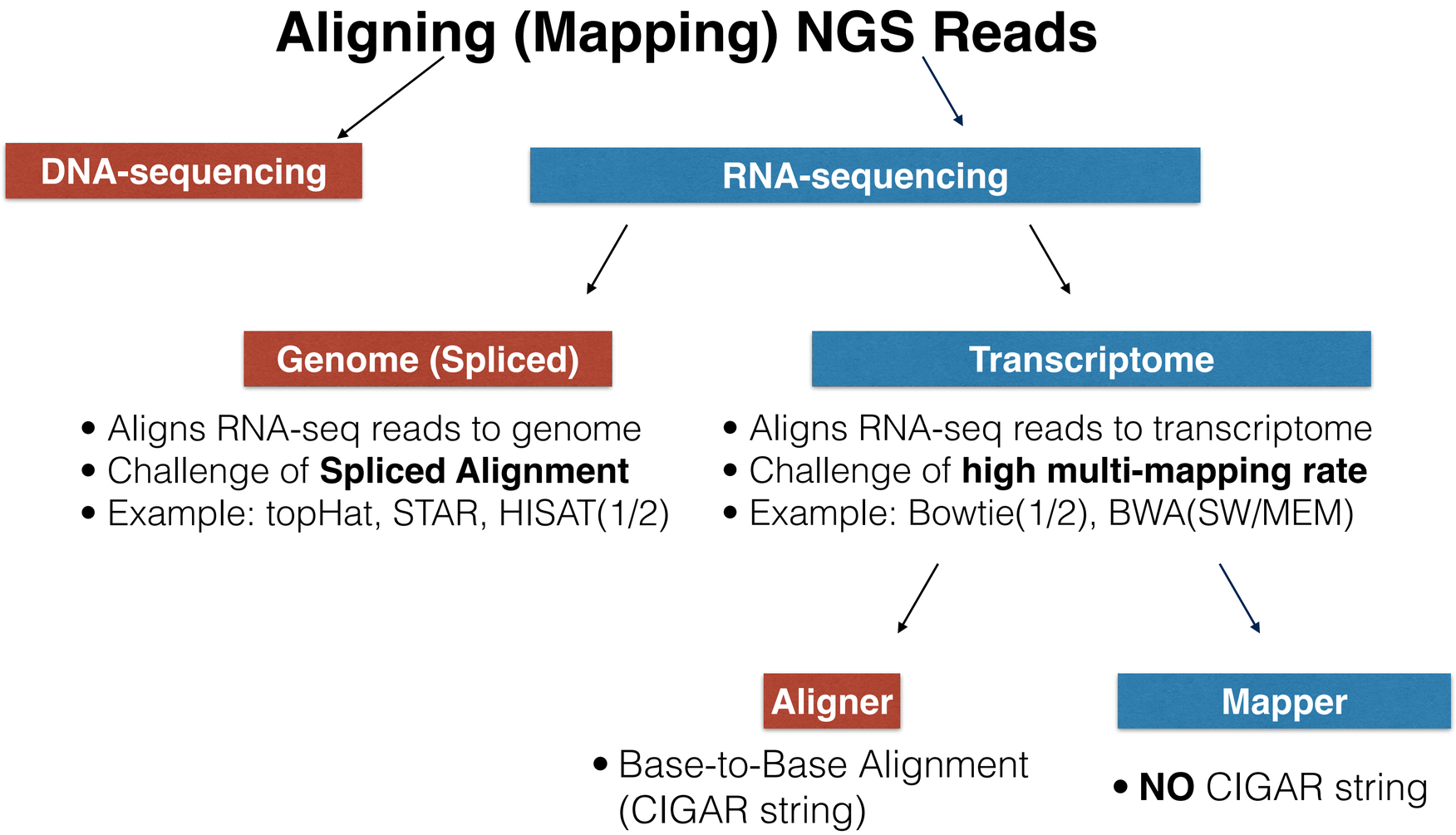
Note: This analysis is a worst-case scenario, where we apply the most naive algorithm possible to solve this problem.

Still, solving the alignment problem as stated above takes too long to be used in practice.

Thus, most tools resort to *heuristics* — approaches that usually work well, but may not return optimal results.

> Specifically, there may be positions on the genome to which a read can be aligned with edit distance $\leq \epsilon$, that are not found (i.e. false negatives). False positives are also possible, but usually resort from finding sub-optimal alignments where better ones exist, and rely on a slightly different formulation of the alignment "problem".
>
> Keep this in mind as we discuss the methods below.

Phylogeny of Read-Alignment



RNA-Seq Read Alignment

Given an RNA-seq read, where *might* it come from?

Two main “regimes”

Align to transcriptome

Align reads directly to txps

No “split” alignments — transcripts contain spliced exons directly.

Typically *a lot* of multi-mapping (80-90% of reads may map to multiple places)

Does *not r*equire target *genome*

Can be used in *de novo* context (i.e. after *de novo* assembly)

Align to genome

Align reads to target genome

Reads spanning exons will be “split” (gaps up to 10s of kb)

Typically little multi-mapping (most reads have single genomic locus of origin)

Requires target *genome*

Can be used to find new transcripts

RNA-Seq Read Alignment

Given an RNA-seq read, where does it come from?

Two main “regimes”

Align to transcriptome

Main computational challenge comes from ubiquitous multi-mapping.

Bowtie

Bowtie 2

BWA

STAR

HISAT (1&2)

…

Align to genome

Main computational challenge comes from spliced alignments.

Top Hat

STAR

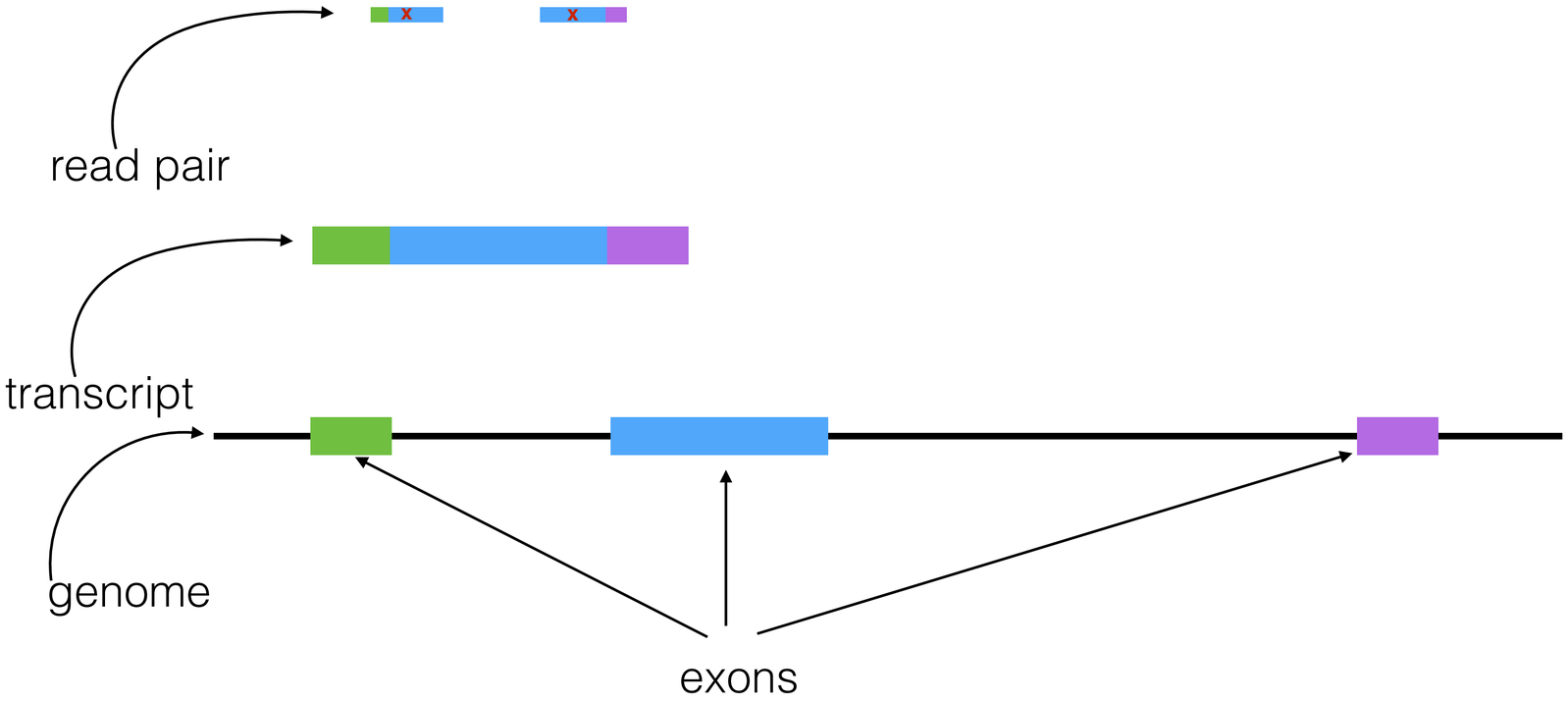
HISAT (1&2)

Map Splice

Subread Aligner

…

Spliced Alignment

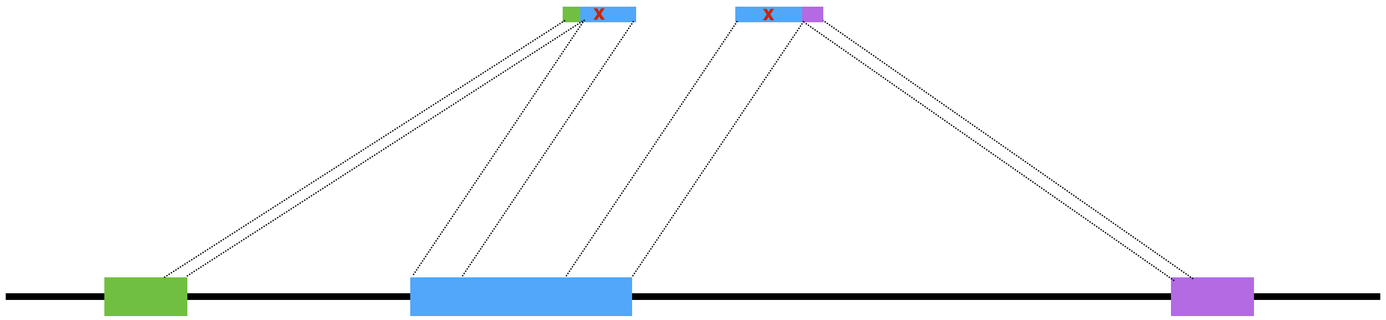


Spliced Alignment

Alight this

back to this

Spliced Alignment



Splice junctions might be known, or *unknown.*

Overlap of read with exon may be *very short*, sequence is ambiguous (e.g. 10 bases).

Sequence of read might be repetitive in the genome.

STAR

BIOINFORMATICS ORIGINAL PAPER

Vol. 29 no. 1 2013, pages 15–21
doi:10.1093/bioinformatics/bts635

Sequence analysis

Advance Access publication October 25, 2012

STAR: ultrafast universal RNA-seq aligner

Alexander Dobin[1,*], Carrie A. Davis[1], Felix Schlesinger[1], Jorg Drenkow[1], Chris Zaleski[1], Sonali Jha[1], Philippe Batut[1], Mark Chaisson[2] and Thomas R. Gingeras[1]

[1]Cold Spring Harbor Laboratory, Cold Spring Harbor, NY, USA and [2]Pacific Biosciences, Menlo Park, CA, USA

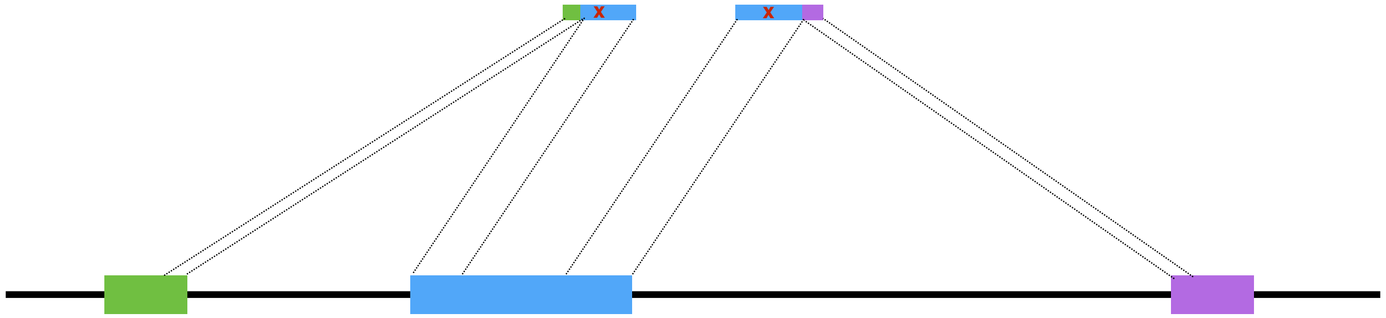
Associate Editor: Inanc Birol

Based on suffix array + prefix-table for seed finding

Custom “chaining” & between match alignment strategy

Capable of both contiguous and spliced alignment, behavior is *highly* configurable via parameters (DNA-seq or RNA-seq alignment directly to the genome)

STAR



Finds long, exact matches using a suffix array.

Potential alignment represented by a chain of MMPs (Maximum Mappable Prefixes).

Alignment validated and processed using dynamic programming algorithm.

STAR

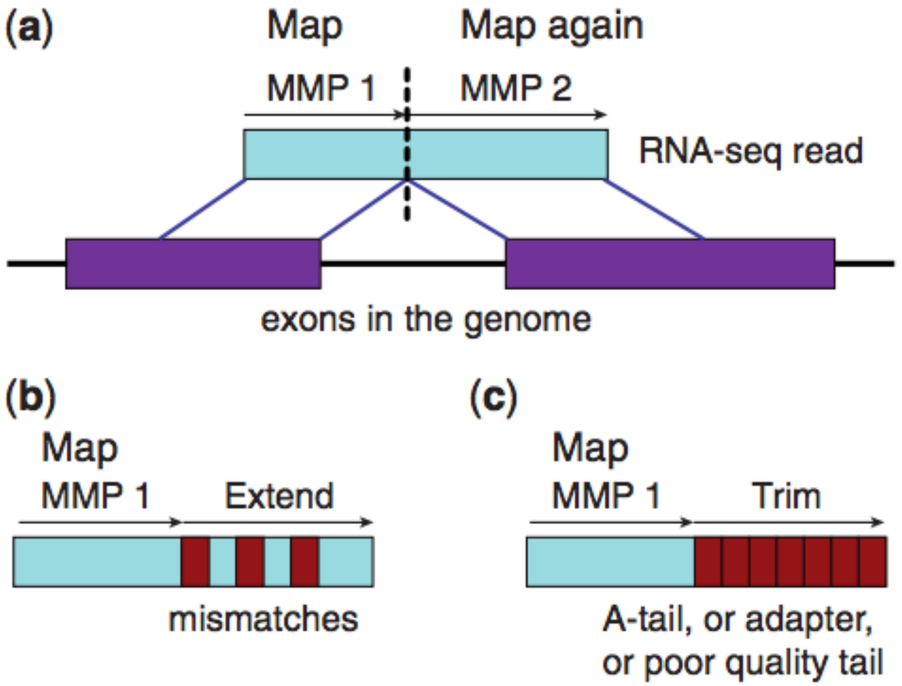


Fig. 1. Schematic representation of the Maximum Mappable Prefix search in the STAR algorithm for detecting (**a**) splice junctions, (**b**) mismatches and (**c**) tails

Figure from: Dobin, Alexander, et al. "STAR: ultrafast universal RNA-seq aligner." Bioinformatics 29.1 (2013): 15-21.

STAR

Accuracy

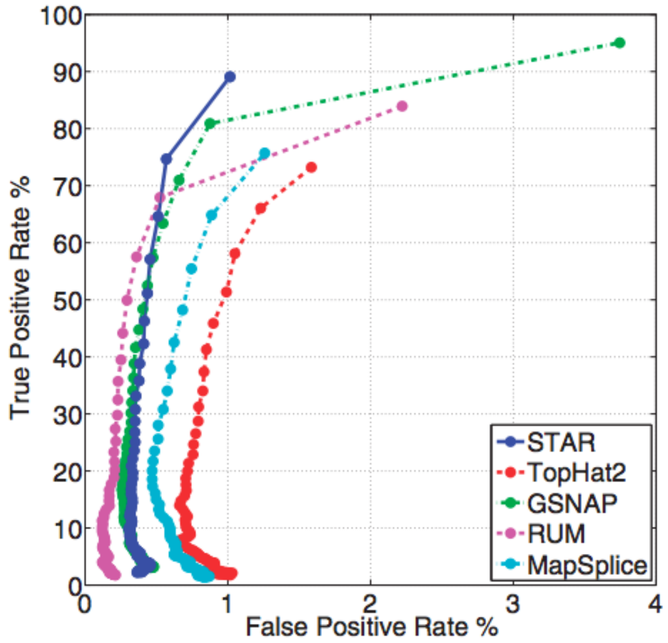


Fig. 2. True-positive rate versus false-positive rate (ROC-curve) for simulated RNA-seq data for STAR, TopHat2, GSNAP, RUM and MapSplice

Speed

Table 1. Mapping speed and RAM benchmarks on the experimental RNA-seq dataset

Aligner	Mapping speed: million read pairs/hour		Peak physical RAM, GB	
	6 threads	12 threads	6 threads	12 threads
STAR	309.2	549.9	27.0	28.4
STAR sparse	227.6	423.1	15.6	16.0
TopHat2	8.0	10.1	4.1	11.3
RUM	5.1	7.6	26.9	53.8
MapSplice	3.0	3.1	3.3	3.3
GSNAP	1.8	2.8	25.9	27.0

Figure from: Dobin, Alexander, et al. "STAR: ultrafast universal RNA-seq aligner." Bioinformatics 29.1 (2013): 15-21.

Modern RNA-Seq *Alignment*

When introduced, STAR was *much* faster than the alternatives. Recently, some new approaches have been developed.

HISAT — *Hierarchical FM-index*

- speed of STAR in an order-of-magnitude less memory
- 1.5-pass alignment to improve align to novel junctions (in STAR now too)

HISAT2 — *Hierarchical Graph FM-index*

- Improved sensitivity
- Very cool / elegant representation of genome, transcripts (etc.)
- Can align to a *population* of genomes

Take home: TopHat (1/2) used to be the *de facto* aligner for RNA-seq -> genome alignment. Now there are *much* better solutions. Also, TopHat 2 has been deprecated in favor of HISAT[1].

1: Pertea, Mihaela, et al. "Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown." Nature Protocols 11, 1650–1667 (2016)

Aligning to the transcriptome

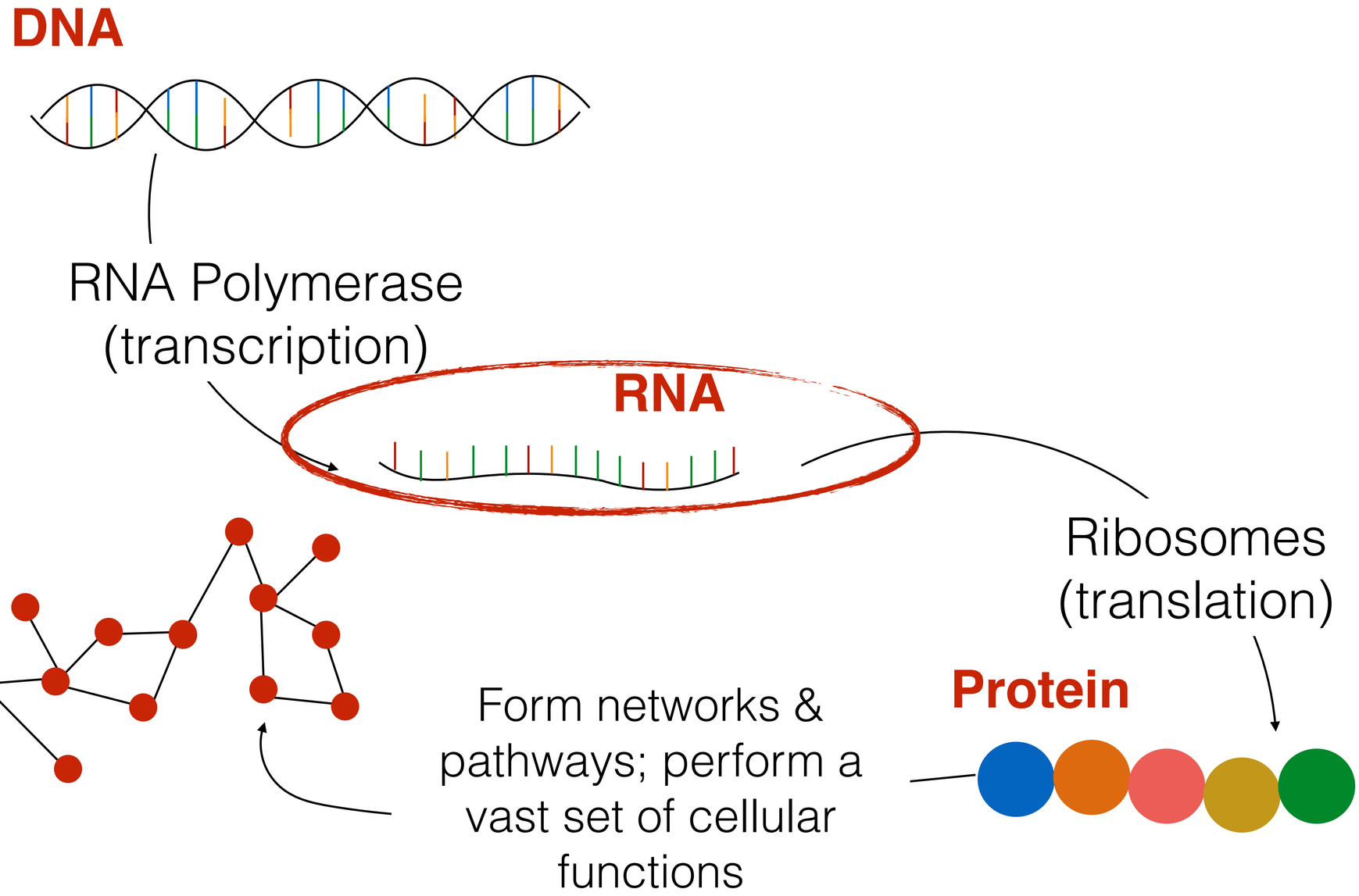
What if we know the transcripts ahead of time?

What if we don't have the target genome (*de novo) txome assembly*?

What if there are other transcripts that we know will be present in our sample, not from the target organism?

One way to handle such cases is to align *directly* to the transcriptome! In this case, we don't need spliced alignment and can use tools like Bowite2/ BWA-MEM and even faster methods (pseudo alignment, quasi-mapping, selective-alignment)

“Flow” of information in the cell



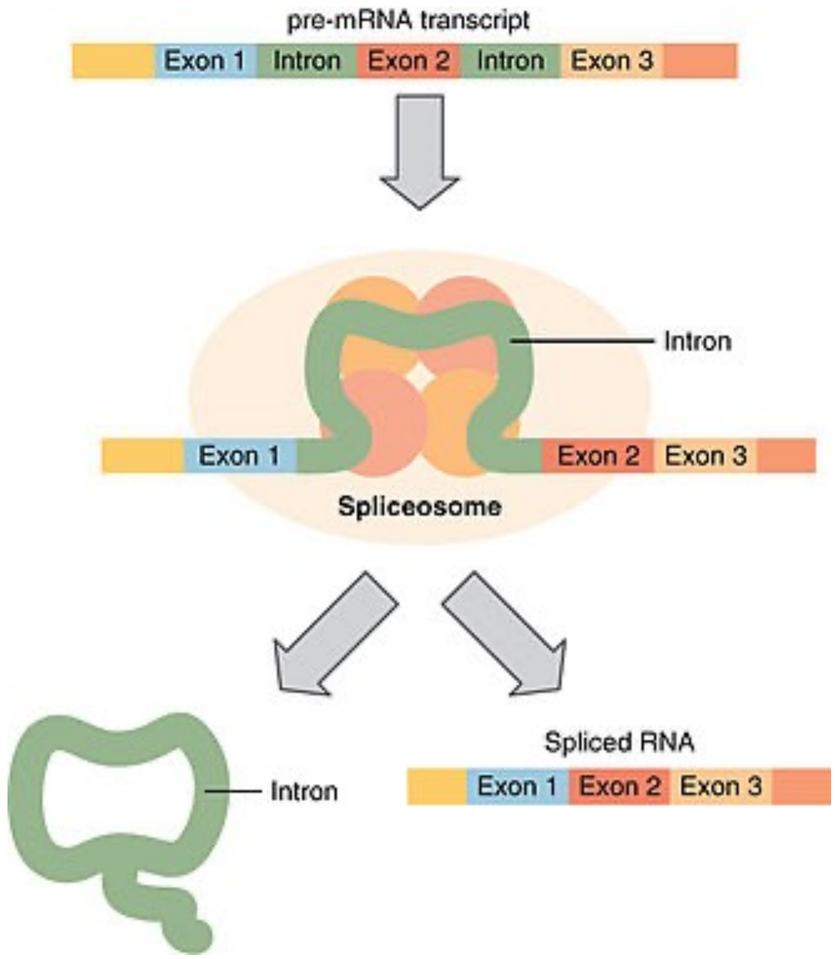
RNA Splicing

DNA transcribed into pre-mRNA

Some “processing occurs”
capping & **polyadenylation**

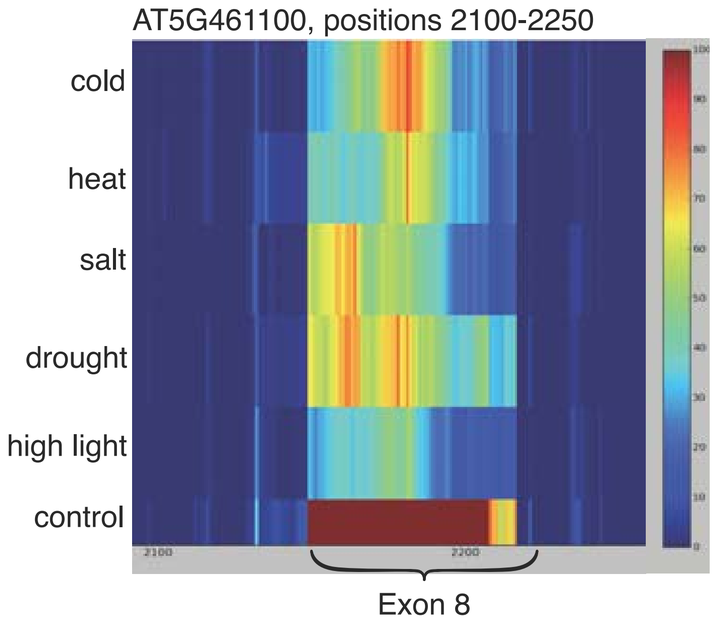
Introns removed from pre-mRNA

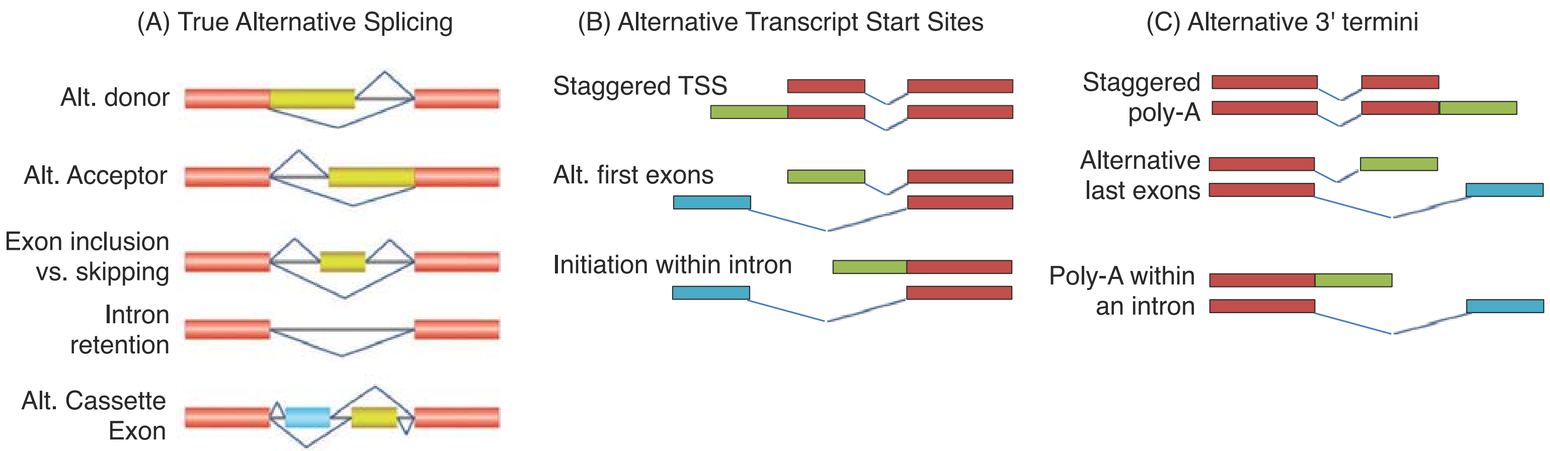
Introns removed resulting in
mature mRNA



Alternative Splicing & Isoform Expression

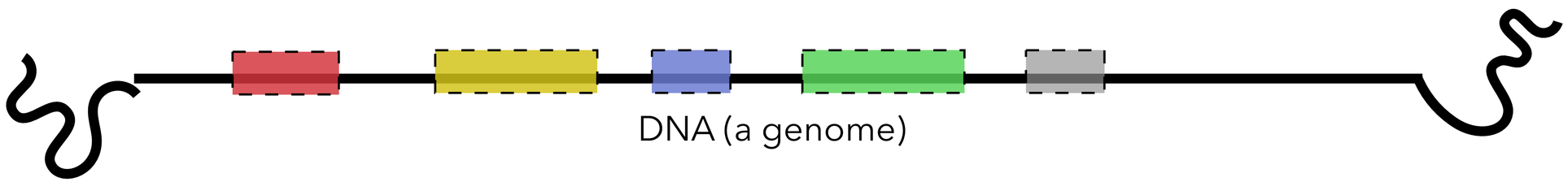
- Expression of genes can be measured via RNA-seq (sequencing transcripts)
- Sequencing gives you short (35-300bp length reads)



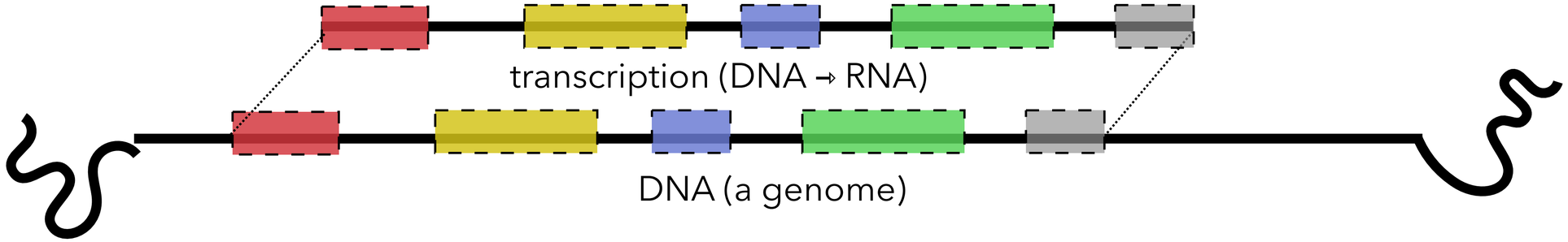


slide courtesy of Carl Kingsford

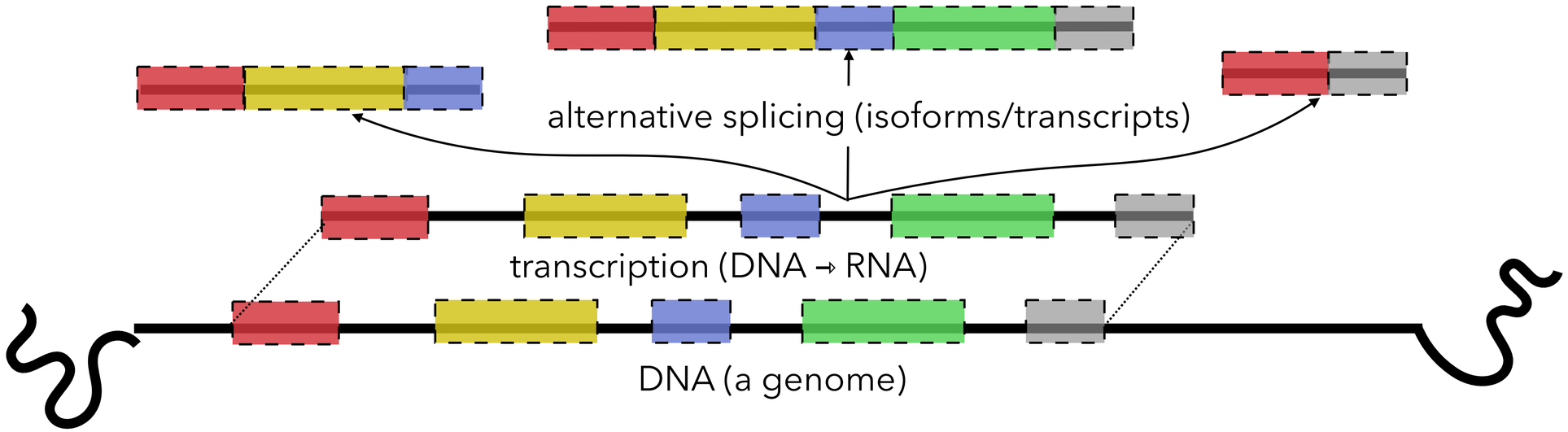
Basics of RNA-seq



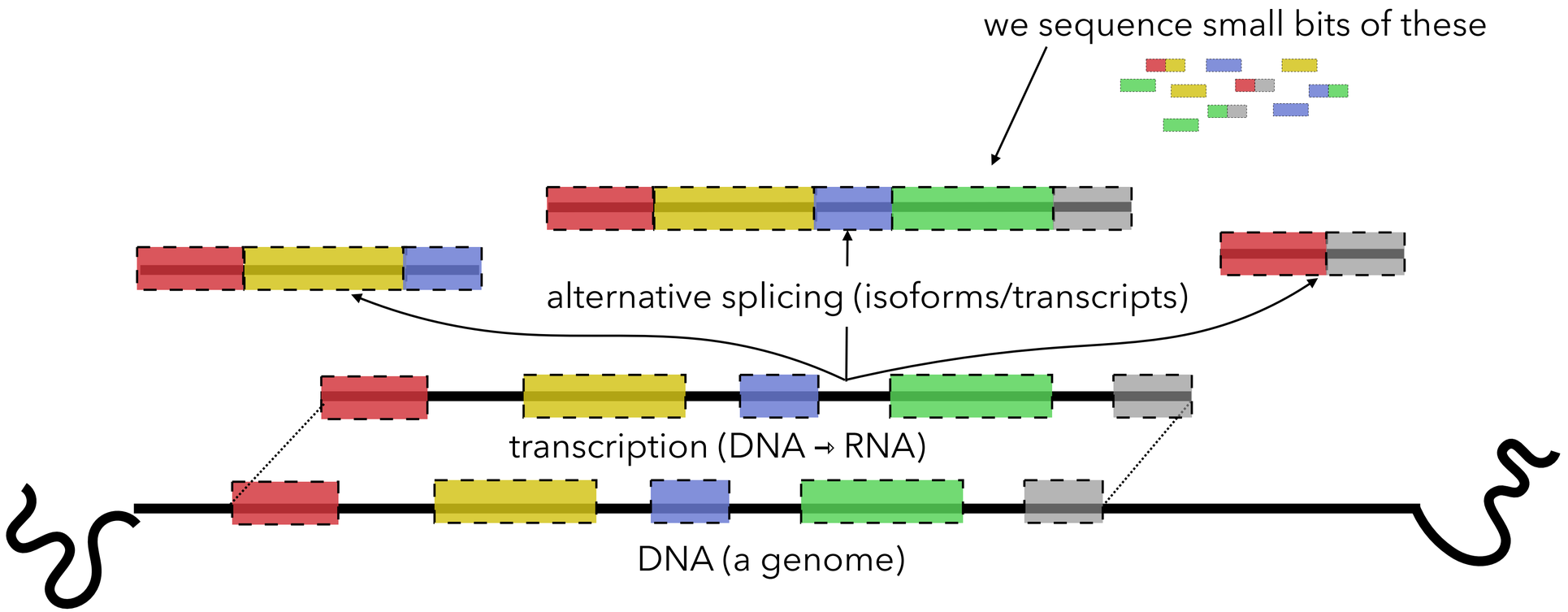
Basics of RNA-seq



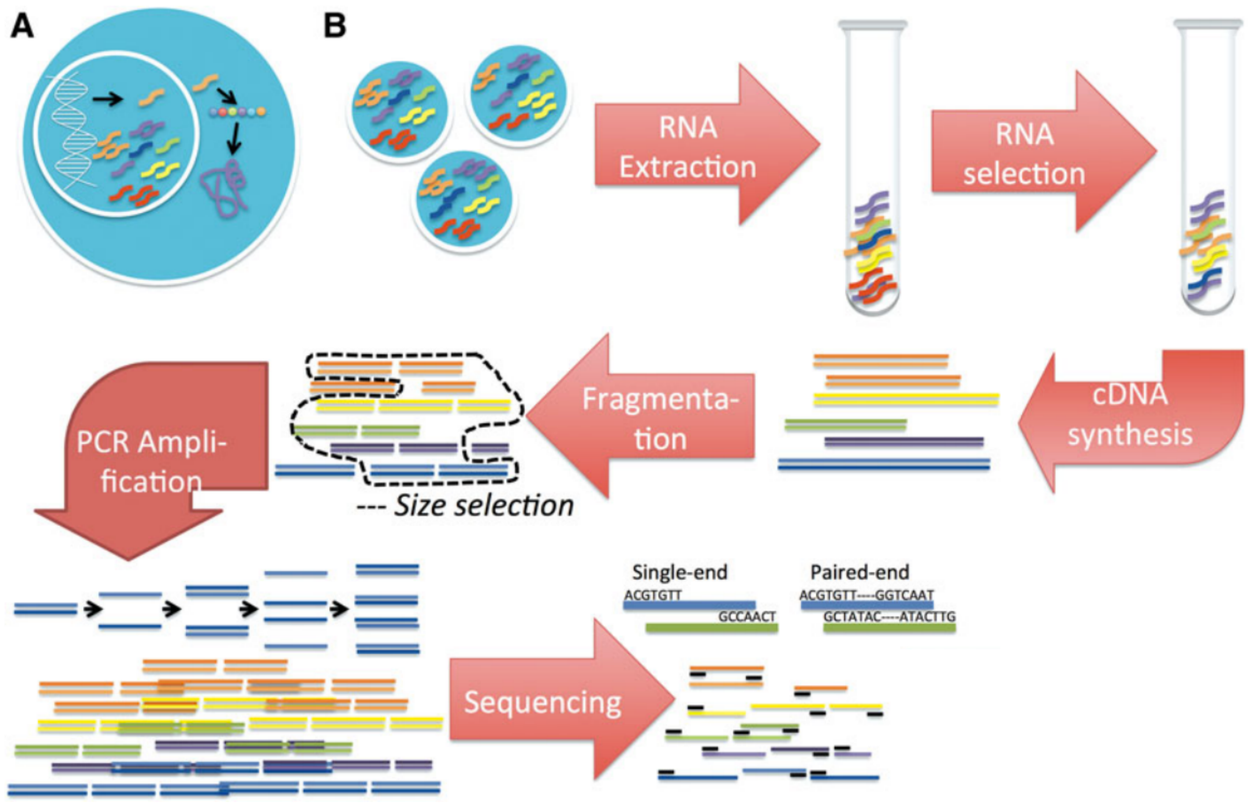
Basics of RNA-seq



Basics of RNA-seq



Actual protocols are much more involved



Prakash, Celine, and Arndt Von Haeseler. "An Enumerative Combinatorics Model for Fragmentation Patterns in RNA Sequencing Provides Insights into Nonuniformity of the Expected Fragment Starting-Point a
of Computational Biology 24.3 (2017): 200-212.

Many uses of RNA-seq for transcriptome analysis

RNA-seq data has many uses in transcriptome analysis. Some common uses include:

Fusion/chimera detection

Variant (SNP, SV, CNV) detection

Transcript assembly

- Genome guided & de novo

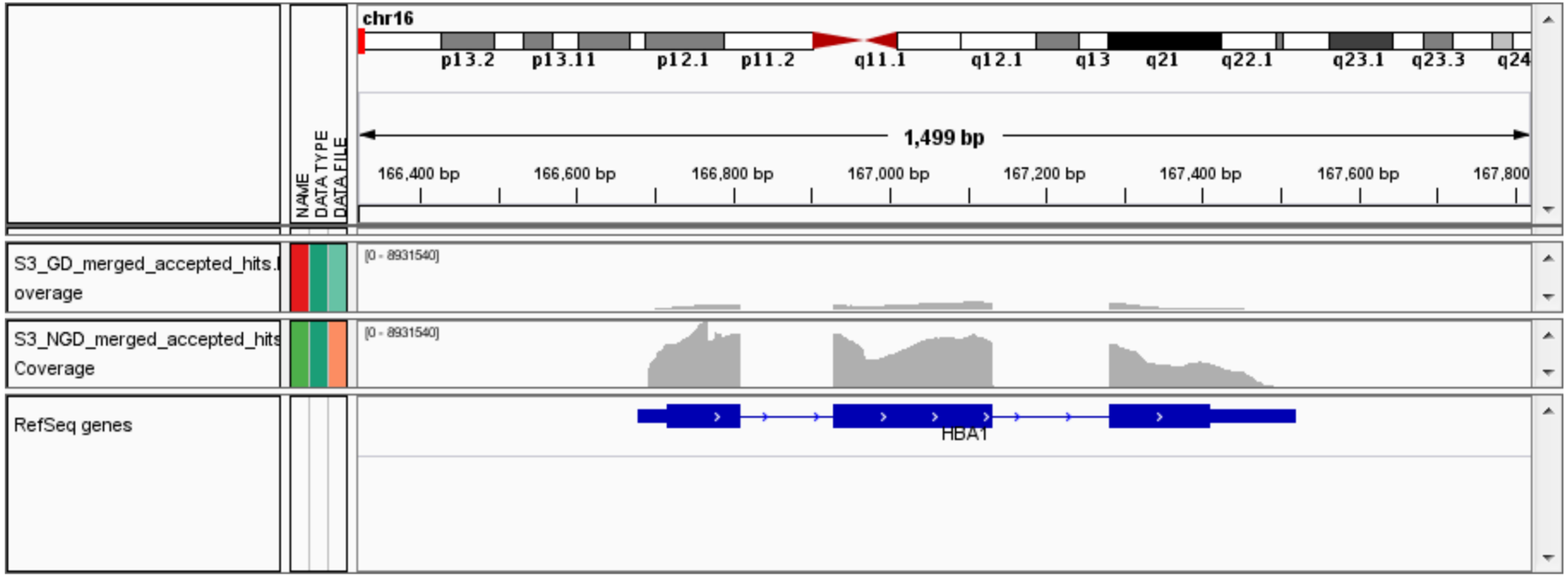
Transcript quantification

- Differential expression, alternative splicing analysis

Build higher-level models of transcription

- co-expression networks -> regulatory networks

The benefit is reads are drawn directly from transcripts!



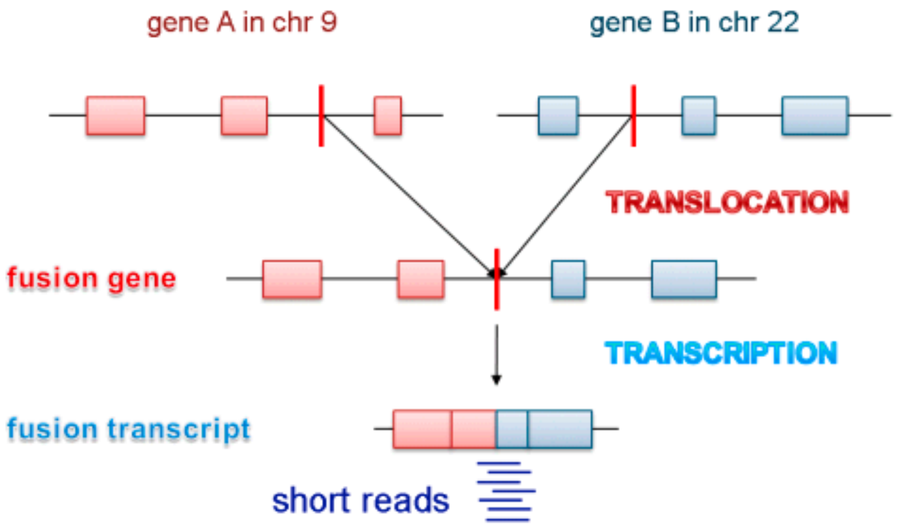
RNA-Seq reads come from a spliced transcript — if we can map them back to the genome, they give us evidence of **transcribed** regions.

Human genome contains > 14,000 pseudogenes [Pei et al. Genome Biology 2012]

Image from: Shin, Heesun, et al. "Variation in RNA-Seq Transcriptome Profiles of Peripheral Whole Blood from Healthy Individuals with and without Globin Depletion." PloS one 9.3 (2014): e91041.

This means you see what is there, not just what is annotated

Fusion/chimera detection



Variant (SNP, SV, CNV) detection

Find small (SNP) or large (SV) variation in how read map back to their genes of origin

Find differences in the number of copies of a gene in the DNA (CNV)

image from:

Many uses of RNA-seq for transcriptome analysis

RNA-seq data has many uses in transcriptome analysis. Some common uses include:

Fusion/chimera detection

Variant (SNP, SV, CNV) detection

Transcript assembly

- Genome guided & de novo

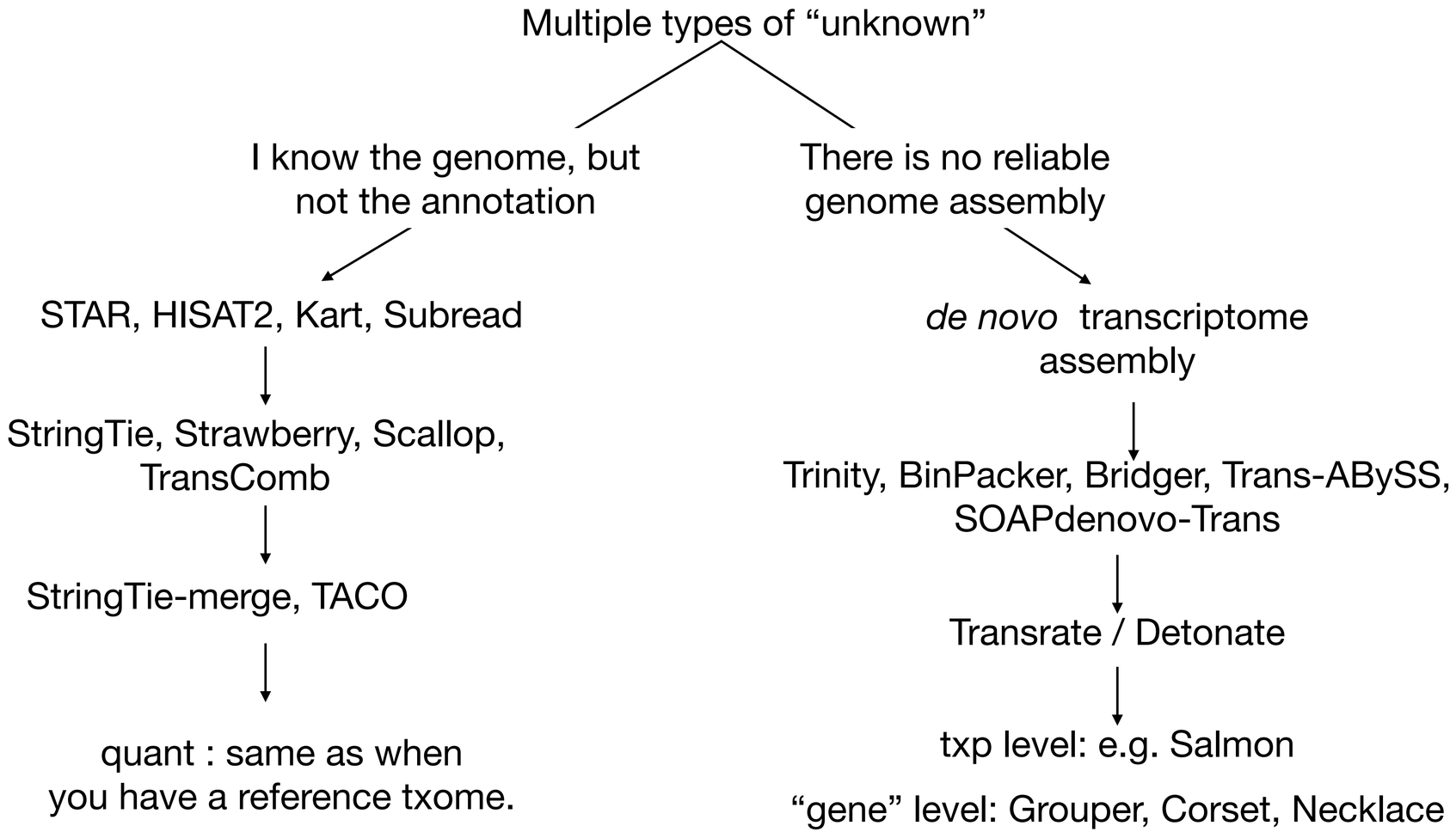
Transcript quantification

- Differential expression, alternative splicing analysis

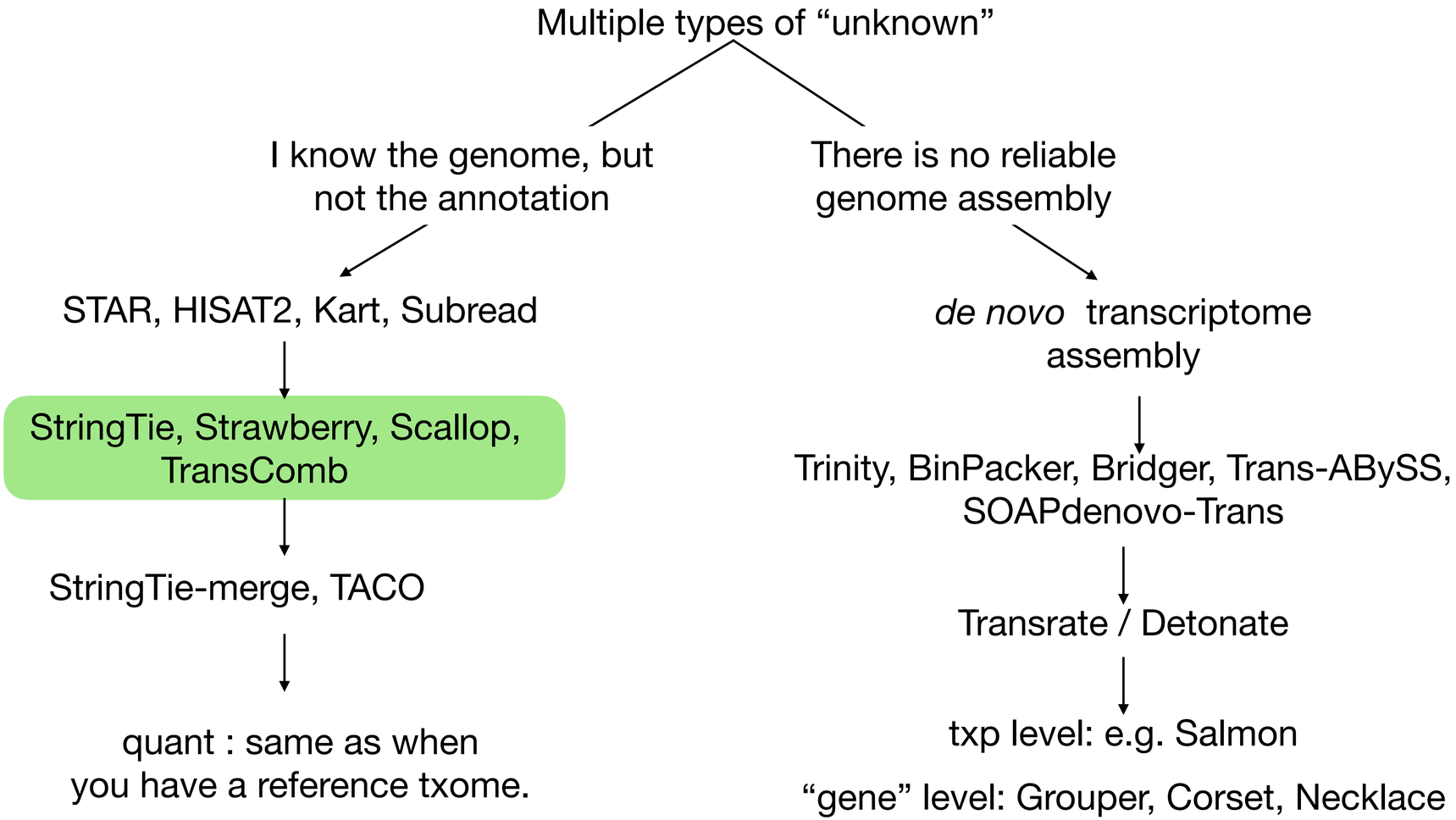
Build higher-level models of transcription

- co-expression networks -> regulatory networks

Into the Unknown : What to do when you don't have / trust your reference

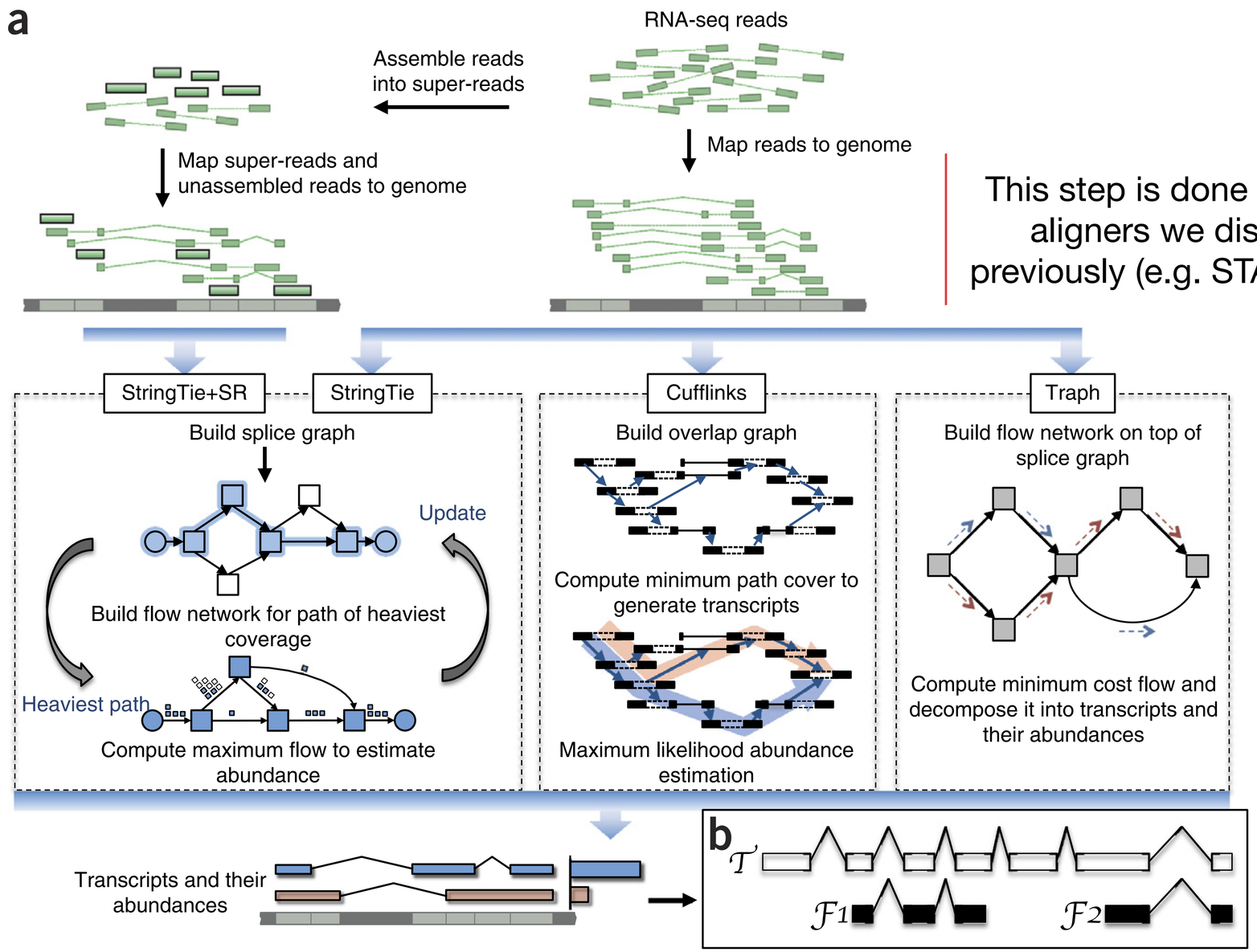


Into the Unknown : What to do when you don't have / trust your reference



I'll focus mostly on *reference-guided* assembly.

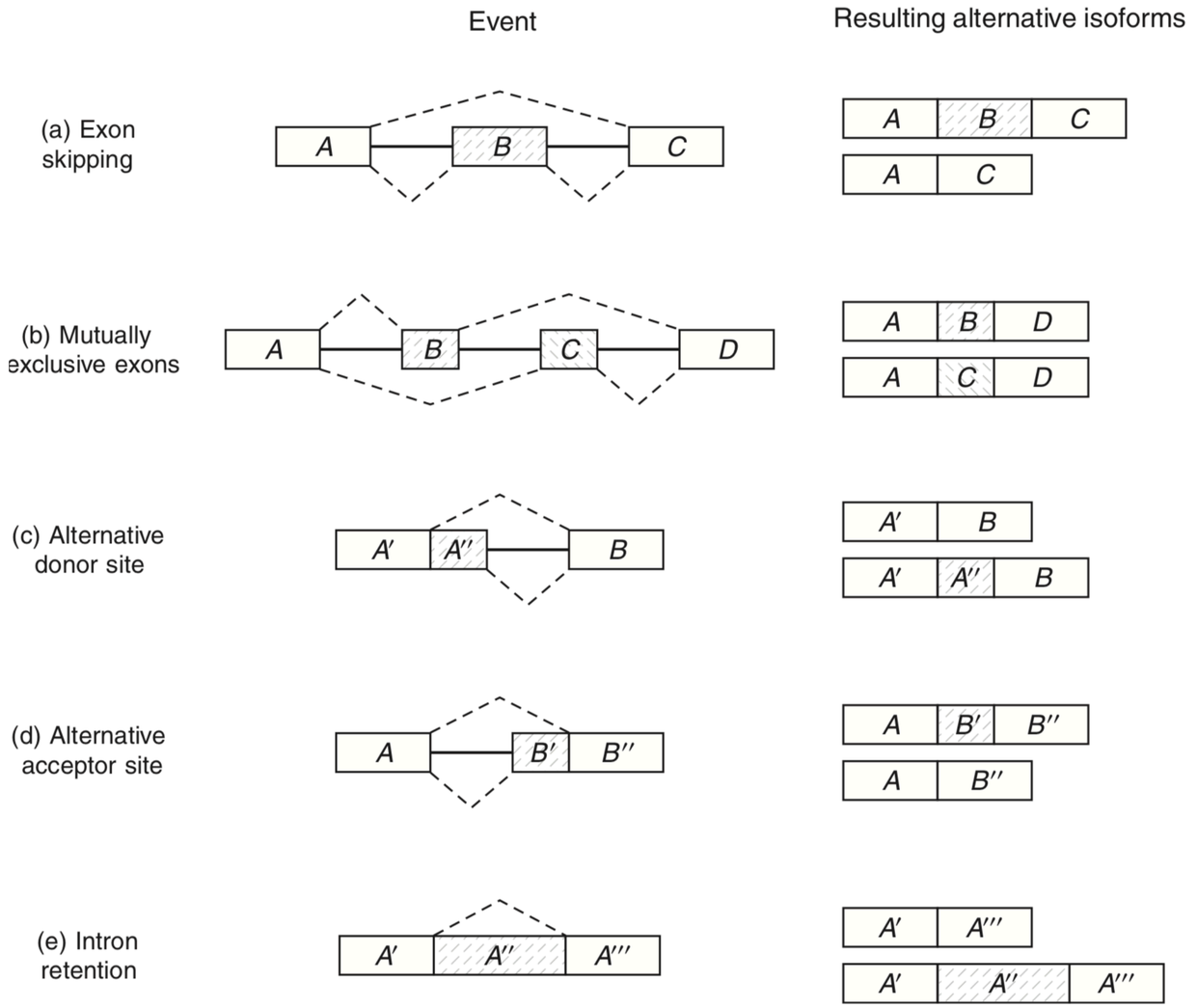
Outline of transcript assembly workflow



This step is done by spliced aligners we discussed previously (e.g. STAR/HISAT2)

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

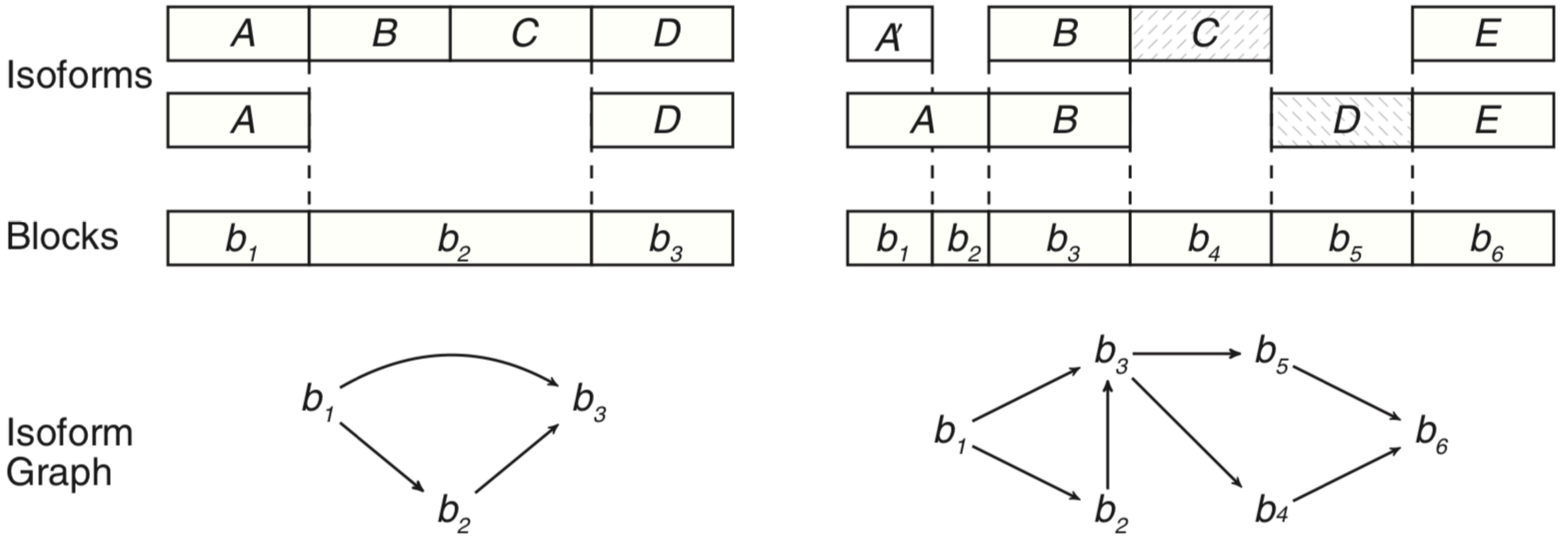
A detour: The splicing graph



Beretta, Stefano, et al. "Modeling alternative splicing variants from RNA-Seq data with isoform graphs." *Journal of Computational Biology* 21.1 (2014): 16-40.

A detour: The splicing graph

In reality we observe coverage by reads, not exons. Therefore, we end up building a slightly different (data-dependent) graph.

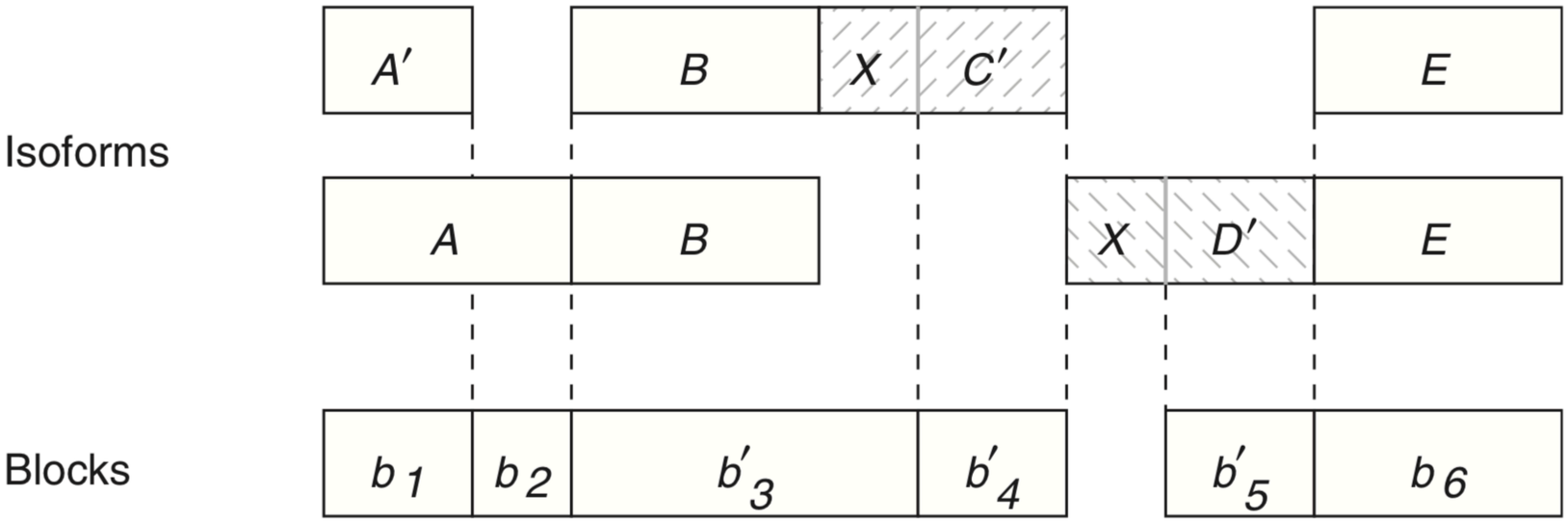


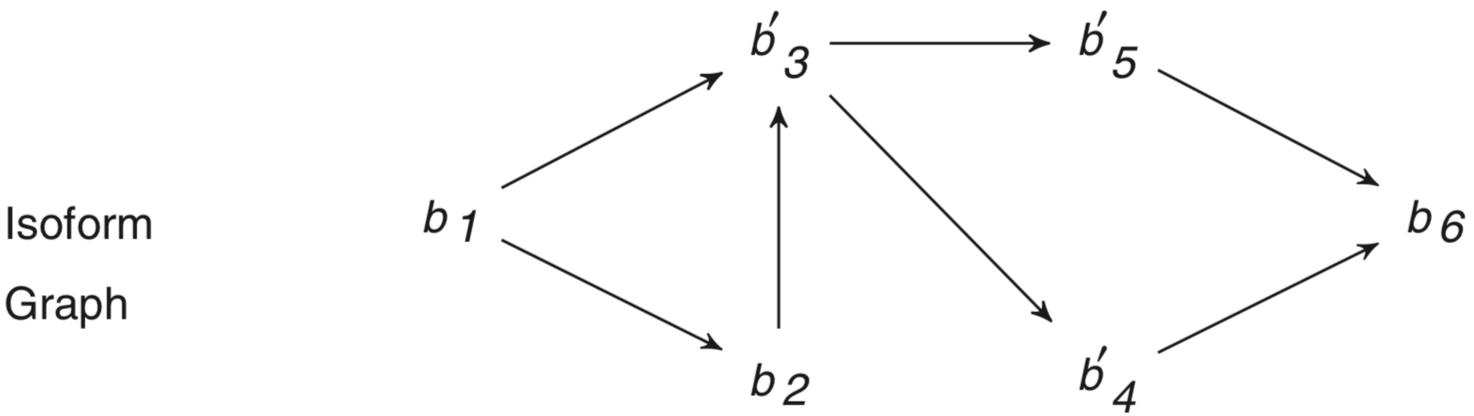
(a) Exon skipping.

(b) Alternative donor sites and mutually exclusive exons.

Beretta, Stefano, et al. "Modeling alternative splicing variants from RNA-Seq data with isoform graphs." *Journal of Computational Biology* 21.1 (2014): 16-40.

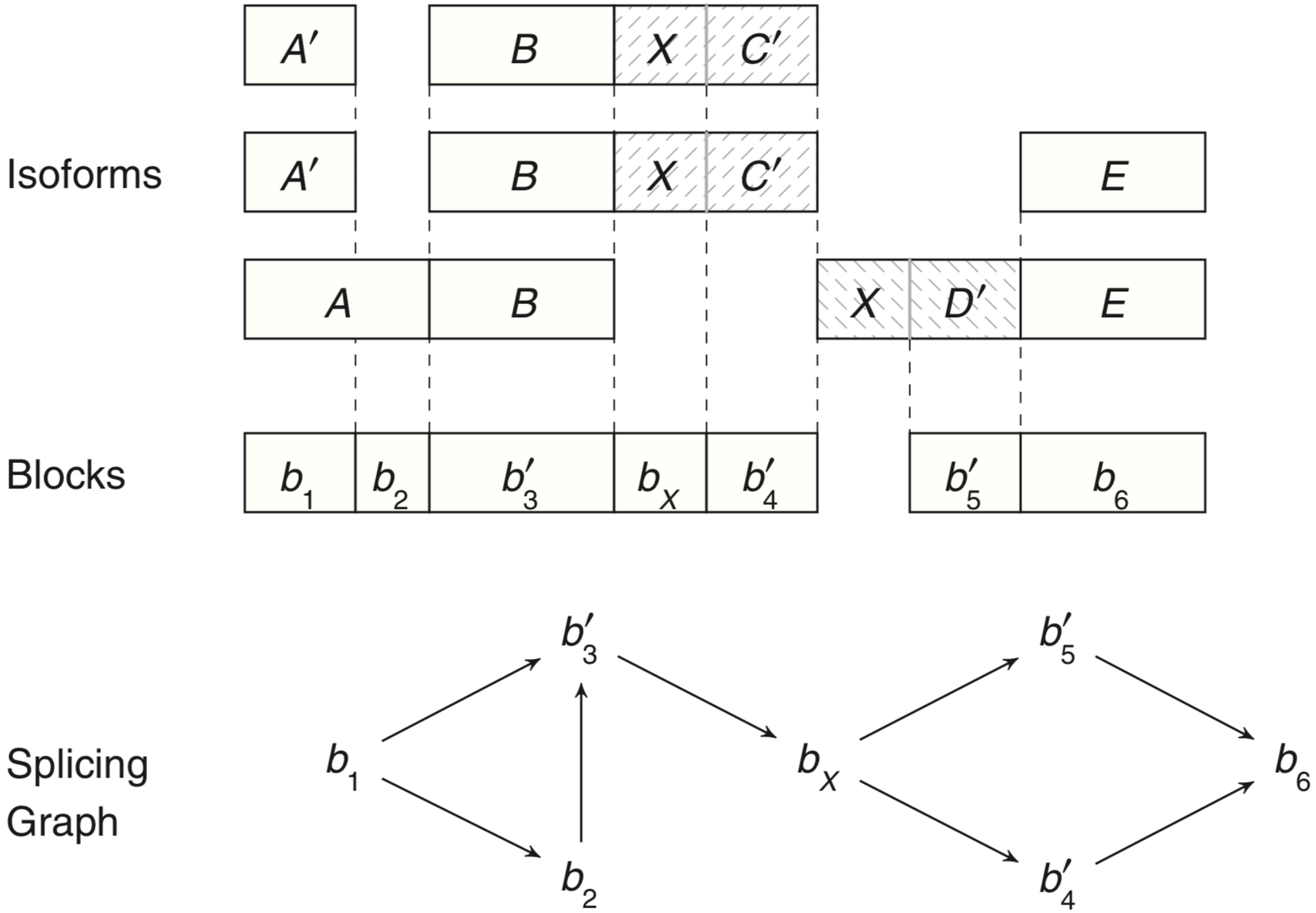
A detour: The splicing graph





Beretta, Stefano, et al. "Modeling alternative splicing variants from RNA-Seq data with isoform graphs." *Journal of Computational Biology* 21.1 (2014): 16-40.

A detour: The splicing graph



Beretta, Stefano, et al. "Modeling alternative splicing variants from RNA-Seq data with isoform graphs." *Journal of Computational Biology* 21.1 (2014): 16-40.

Building of Splice Graph

StringTie builds an alternative splicing graph (ASG) from all reads at a genomic locus.

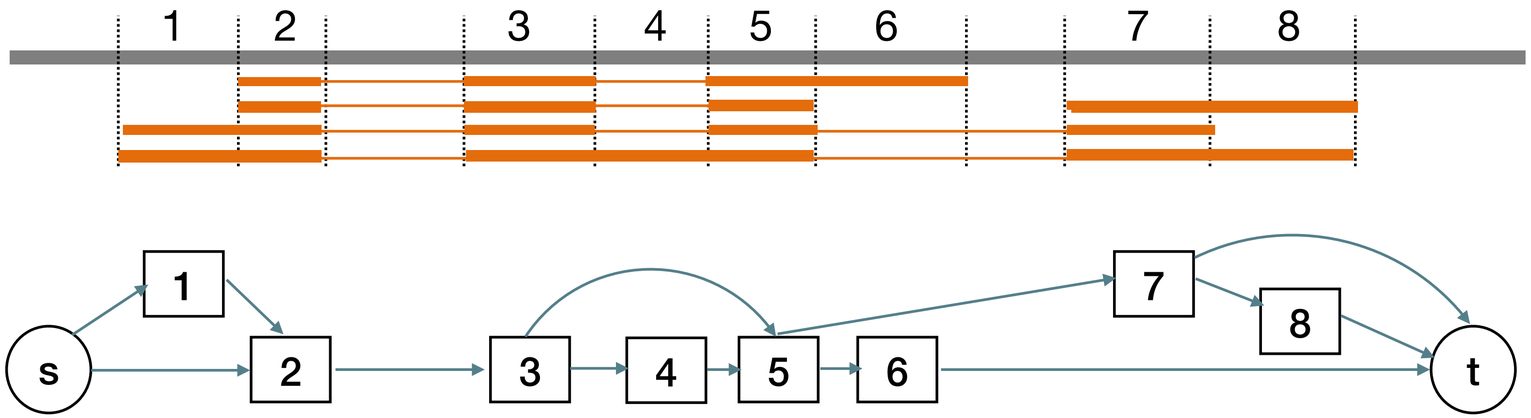
Skips this locus if too many (>95%) of the reads here are multi-mapping

Otherwise, reads are naively given 1/k mass at each of their k multi-mapping loci.

Splice graph is a DAG where nodes are contiguous genomic regions not interrupted by spliced alignments, and edges are placed between two nodes between which a spliced alignment occurs.

Building of Splice Graph

ASG example (adapted from supp fig. 1)



Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

Processing the Splice Graph

StringTie identifies transcripts using the ASG with the following iterative process:

1. Heuristically choose a “heavy” path (a path with the heaviest node) in the ASG

2. Estimate path expression by computing max-flow in a flow graph corresponding to this sub-path of the ASG. Subtract the read mass assigned to the nodes in this path & repeat.

Choosing a Heaviest Path

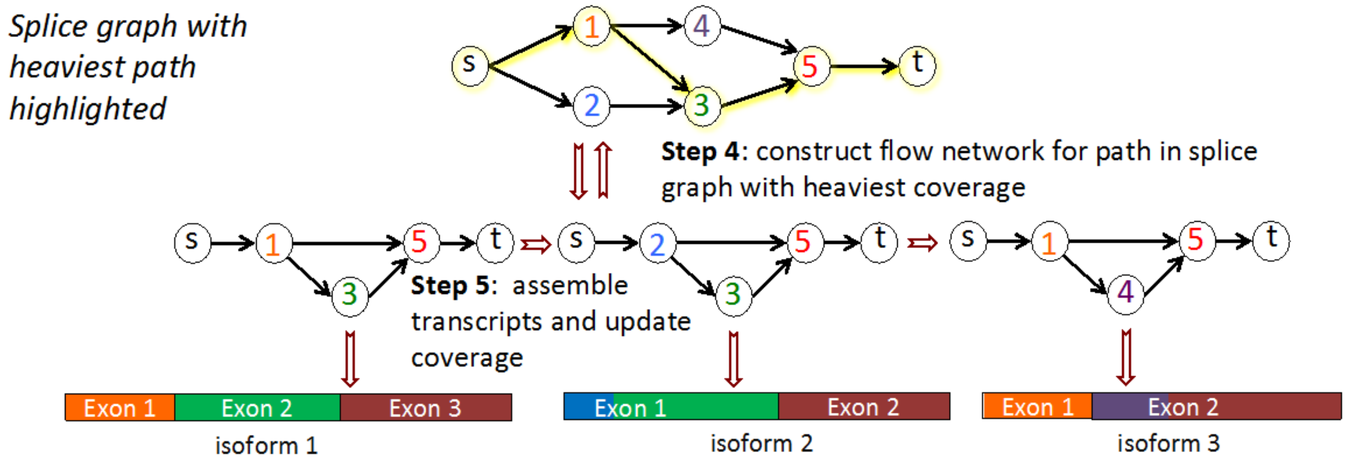
StringTie chooses the heaviest path greedily, as follows (this is an O(n) algorithm):

Pick the ASG node with the highest per-base read coverage.

Extend nodes to the **source** by adding to the path the adjacent node with the largest # of compatible read fragments.

Extend nodes to the **sink** by adding to the path the adjacent node with the largest # of compatible read fragments.

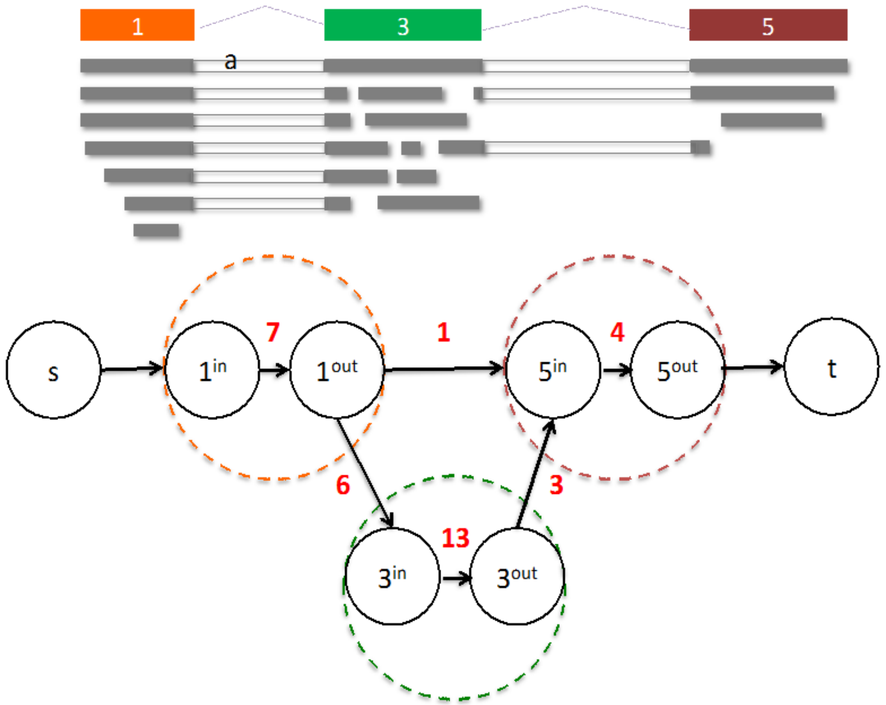
Constructing the Flow Network



Note: The flow network is constructed separately for each selected transcript — the network on which the flow problem is solved does *not* correspond to the entire ASG!

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

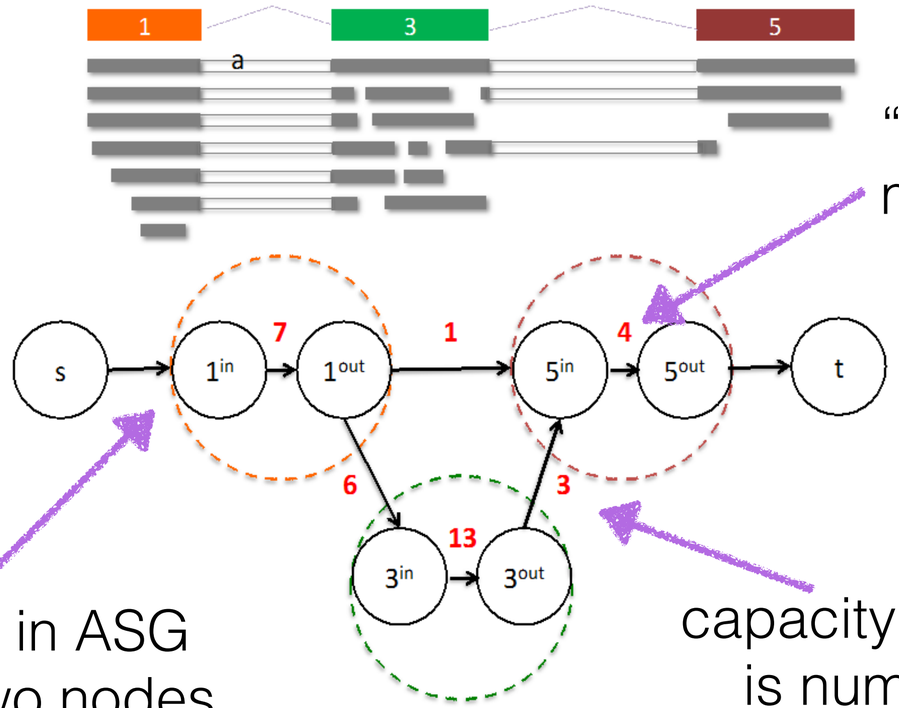
Constructing the Flow Network



Supplementary Figure 13. Flow network associated with a transcript (shown with colored nodes). 15 fragments (shown in grey) align to the transcript. Two nodes in the flow network are connected iff a fragment starts and ends at those nodes. E.g., nodes 1 and 5 are connected because fragment (a) starts at node 1 and ends at node 5. For each colored node in the transcript, two nodes are created in the flow network. Capacities on edges (not connecting source or sink) are shown in red.

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

Constructing the Flow Network



"within" edges may also have a bias factor

$$b_v = \frac{\sum_{u \in ASG} w(f_{uv})}{\sum_{u \in ASG} w(f_{vu})}$$

each node in ASG becomes two nodes in / out in flow network

capacity of "between" edges is number of alignments starting in one node and ending in the other

Supplementary Figure 13. Flow network associated with a transcript (shown with colored nodes). 15 fragments (shown in grey) align to the transcript. Two nodes in the flow network are connected iff a fragment starts and ends at those nodes. E.g., nodes 1 and 5 are connected because fragment (a) starts at node 1 and ends at node 5. For each colored node in the transcript, two nodes are created in the flow network. Capacities on edges (not connecting source or sink) are shown in red.

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

Constructing the Flow Network

Given the flow network, StringTie solves a *generalized* max-flow problem

generalized — edges may have multipliers (bias factors), so flow may be gained or lost as it is sent through the network.

$N = (G, s, t, c)$, where $G = (V, E)$, with $E \subseteq V \times V$, ← Flow Network

$s \in V$ and $t \in V$ ← Source / Sink

$c : E \to R^+$ ← Edge capacities

$f : E \to R^+$ ← Find a **flow** that is **maximum**

$||f|| = \Sigma_{(v, t) \in E} f((v, t))$ ← Find a **flow** that is **maximum**

(1) $0 \leq f((u, v)) \leq c((u, v))$, for every $(u, v) \in E$,
(2) $\Sigma_{(u, v) \in E} f((u, v)) = \Sigma_{(v, u) \in E} f((v, u))$, for every $v \in V$, $v \neq s, t$.

Satisfying

$l_{(u,v)} \leq f((u,v)) \leq c((u,v)), \quad \text{for every } (u,v) \in E$ ← Satisfying

Constructing the Flow Network

Given the flow network, StringTie solves a *generalized* max-flow problem

generalized — edges may have multipliers (bias factors), so flow may be gained or lost as it is sent through the network.

$N = (G, s, t, c)$, where $G = (V, E)$, with $E \subseteq V \times V$ ← Flow Network

$s \in V$ and $t \in V$ ← Source / Sink

$c : E \to R^+$ ← Edge capacities

$f : E \to R^+$ ← **maximum flow** that is **maximum**

$||f|| = \Sigma_{(v, t) \in E} f((v, t))$

(1) $0 \leq f((u, v)) \leq c((u, v))$, for every $(u, v) \in E$,

(2) $\Sigma_{(u, v) \in E} f((u, v)) = \Sigma_{(v, u) \in E} f((v, u))$, for every $v \in V$, $v \neq s, t$. ← Satisfying

$l_{(u,v)} \leq f((u,v)) \leq c((u,v)), \quad$ for every $(u,v) \in E$

StringTie solves this generalized max-flow problem using an augmenting path algorithm

Recall: Max Flow

Flow network: G = (V, E, s, t, c)

Find a flow $f : E \rightarrow R^+$ of maximum value

Subject to:

(1) Capacity : $0 \leq f(e) \leq c_e$ for all $e \in E$

(2) Conservation : For every $v \in V$ (apart from s and t)

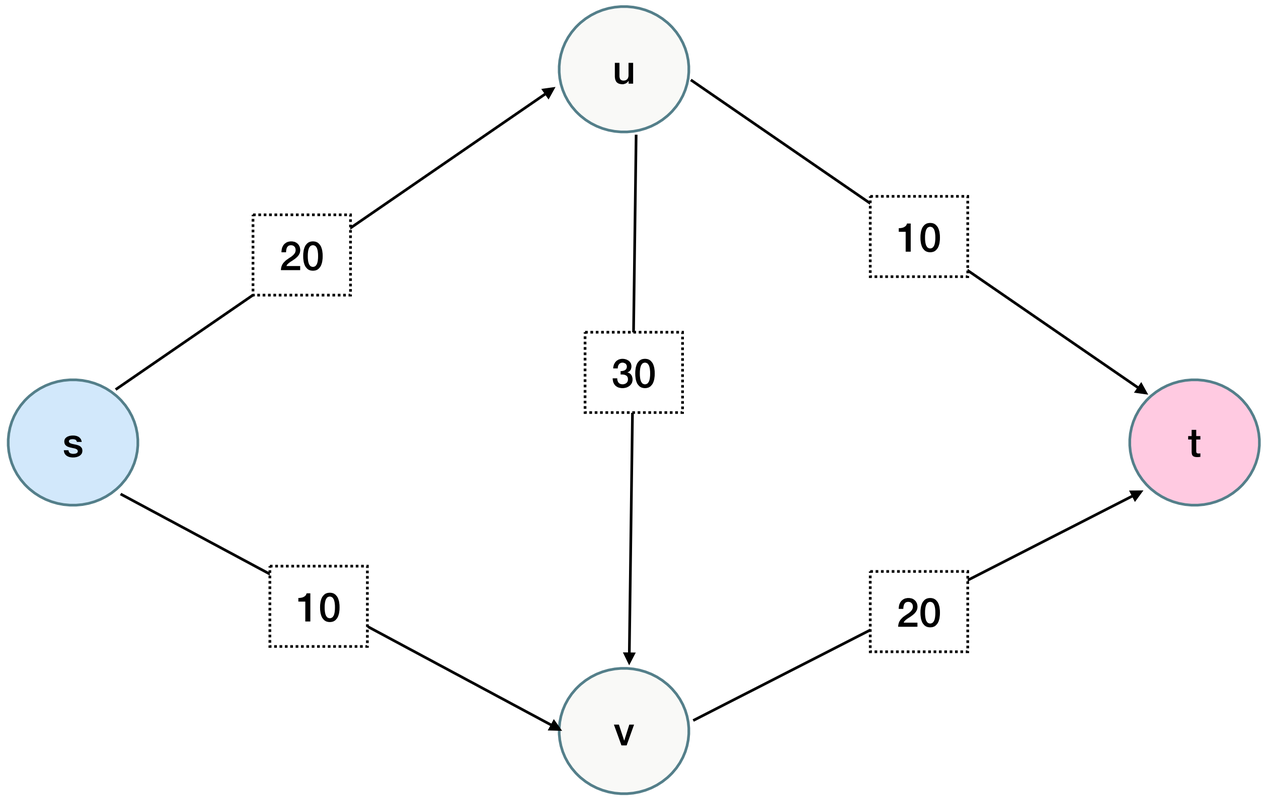
$$\sum_{e \text{ into } v} f(e) = \sum_{e' \text{ out of } v} f(e')$$

Value of the flow is given by:

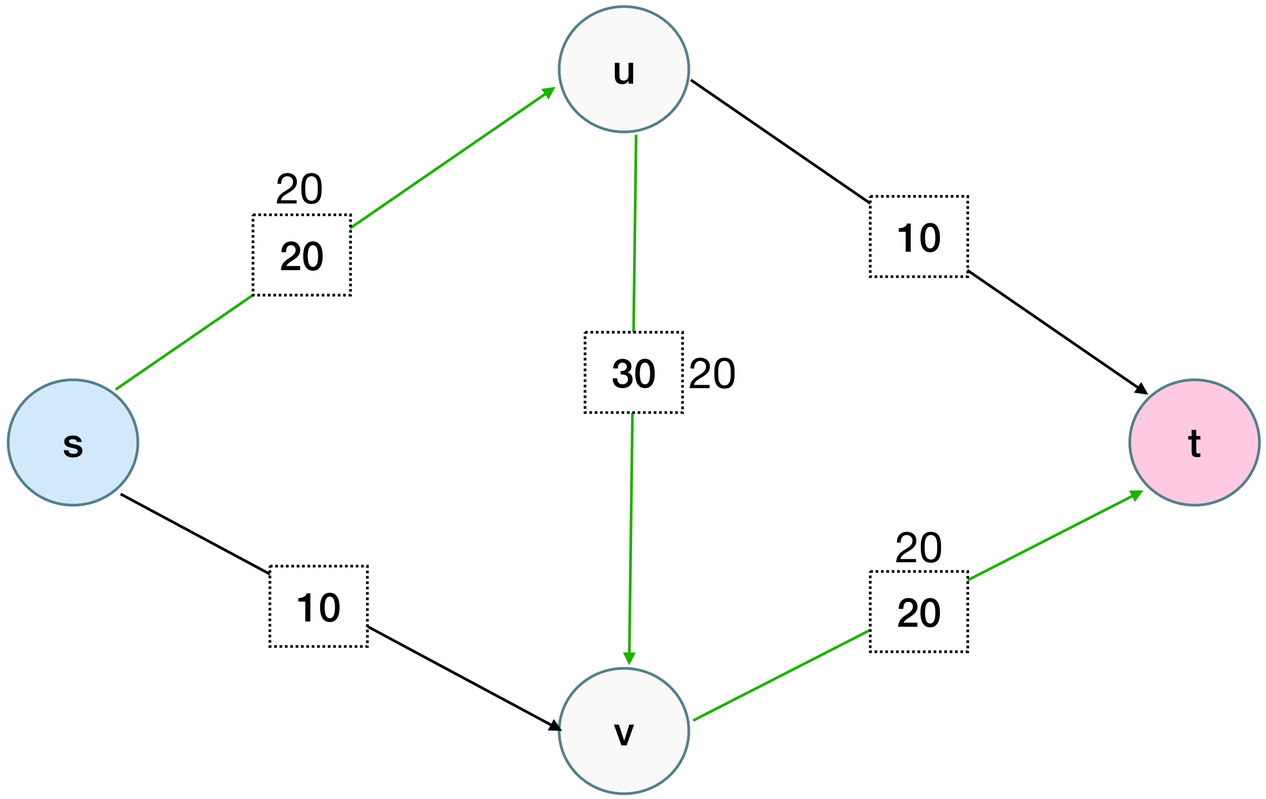
$$v(f) = \sum_{e \textbf{ out of } s} f(e)$$

The below slides follow 7.1 in Kleinberg & Tardos

Recall: Basic Algorithm

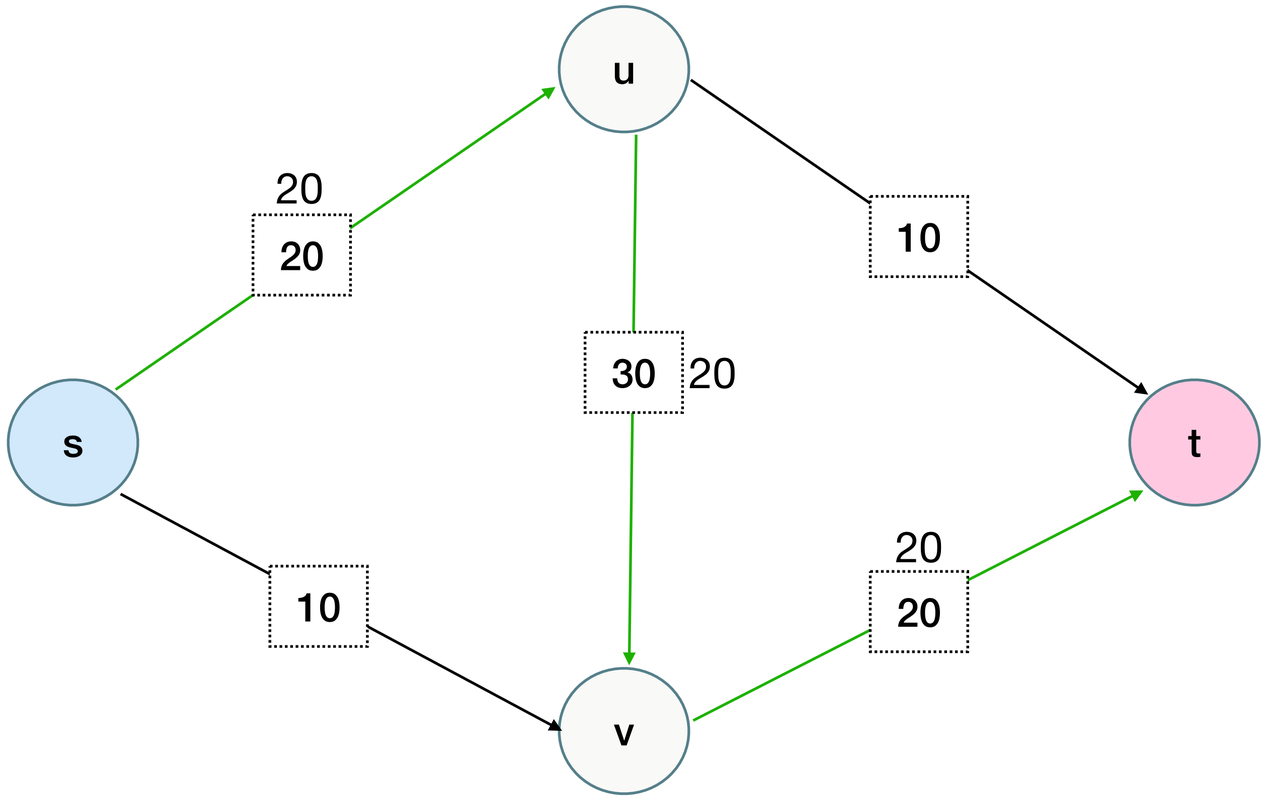


Recall: Basic Algorithm



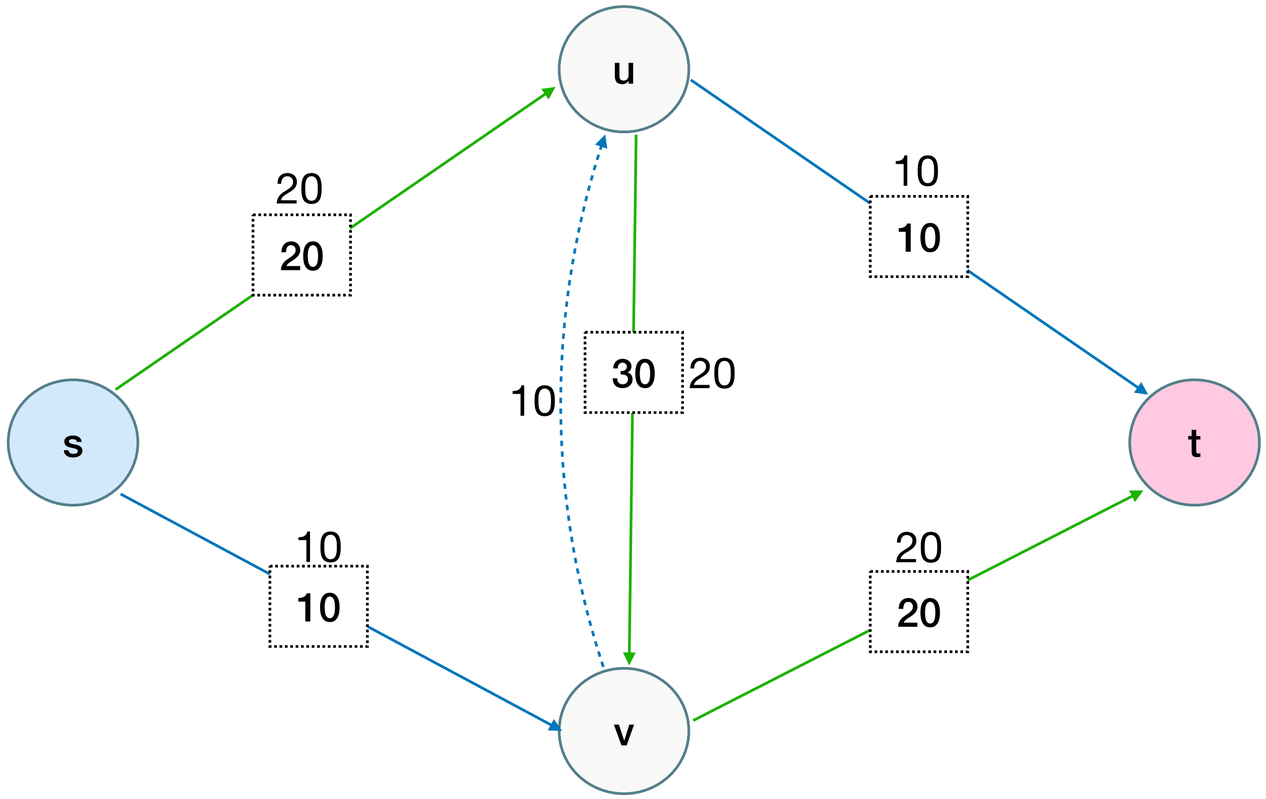
This achieves a flow of value 20, which respects (1) and (2). Is it maximum?

Recall: Basic Algorithm



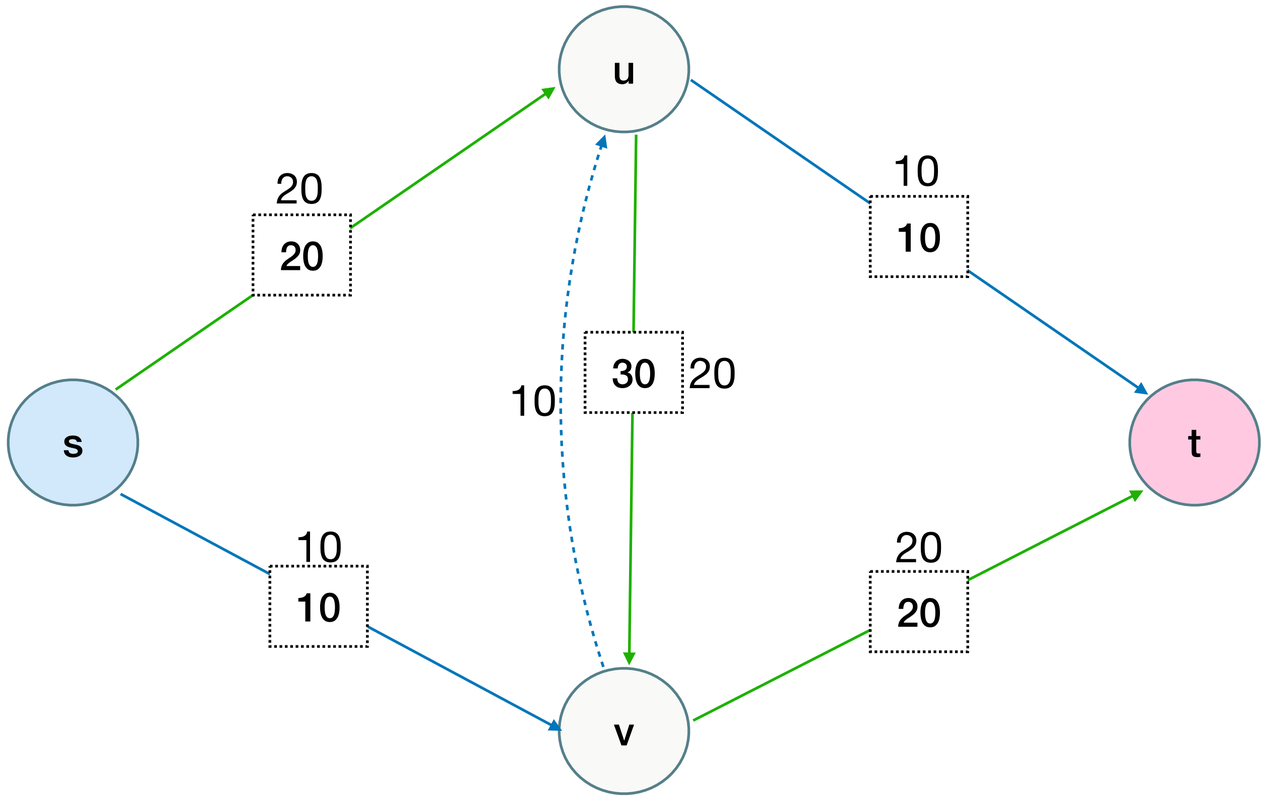
This achieves a flow of value 20, which respects (1) and (2). Is it maximum?
No, but now we are “stuck” by the edges we chose. What if we could “undo” some of the flow?

Recall: Basic Algorithm



This achieves a flow of value 20, which respects (1) and (2). Is it maximum? No, but now we are “stuck” by the edges we chose. What if we could “undo” some of the flow?

Recall: Basic Algorithm



The “dotted” line here is a “backward” edge — it doesn’t exist in the original graph. But flows realized using such residual edges can always be realized in the original graph by changing the forward flows. This leads to the formal idea of the residual graph.

Recall: Basic Algorithm

If G is a flow network with a valid flow f, then the residual

Graph G_f:

Has the same node set as G

for each e = (u,v) in G where f(e) < c_e, G_f has an edge e=(u,v) with capacity given by c_e - f(e).

for each e = (u,v) in G where f(e) > 0, G_f has an edge e'=(v,u) with capacity f(e) — these are "backward edges"

Recall: Basic Algorithm

Let **bottleneck**(P,f) for a simple s-t path P with flow f be the minimum residual capacity of any edge on P. We define the following subroutine:

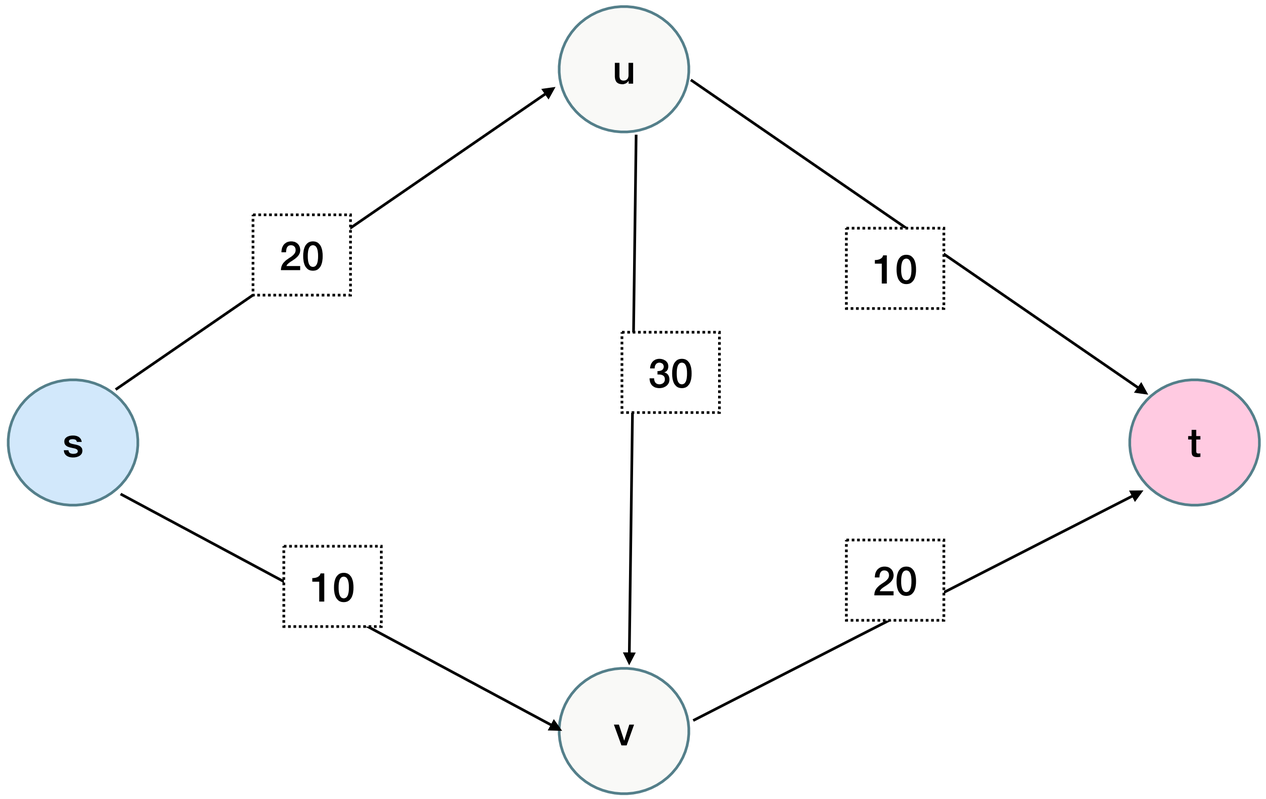
```
augment(f,P)
 let b = bottleneck(P,f)
 for e = (u,v) in P
  if e = (u,v) is forward
    increase f(e) in G by b
  else (u,v) is backward let e = (v,u)
    decrease f(e) in G by b
  endif
 endfor
 return f
```

Recall: Basic Algorithm

We can then find a maximum flow as follows

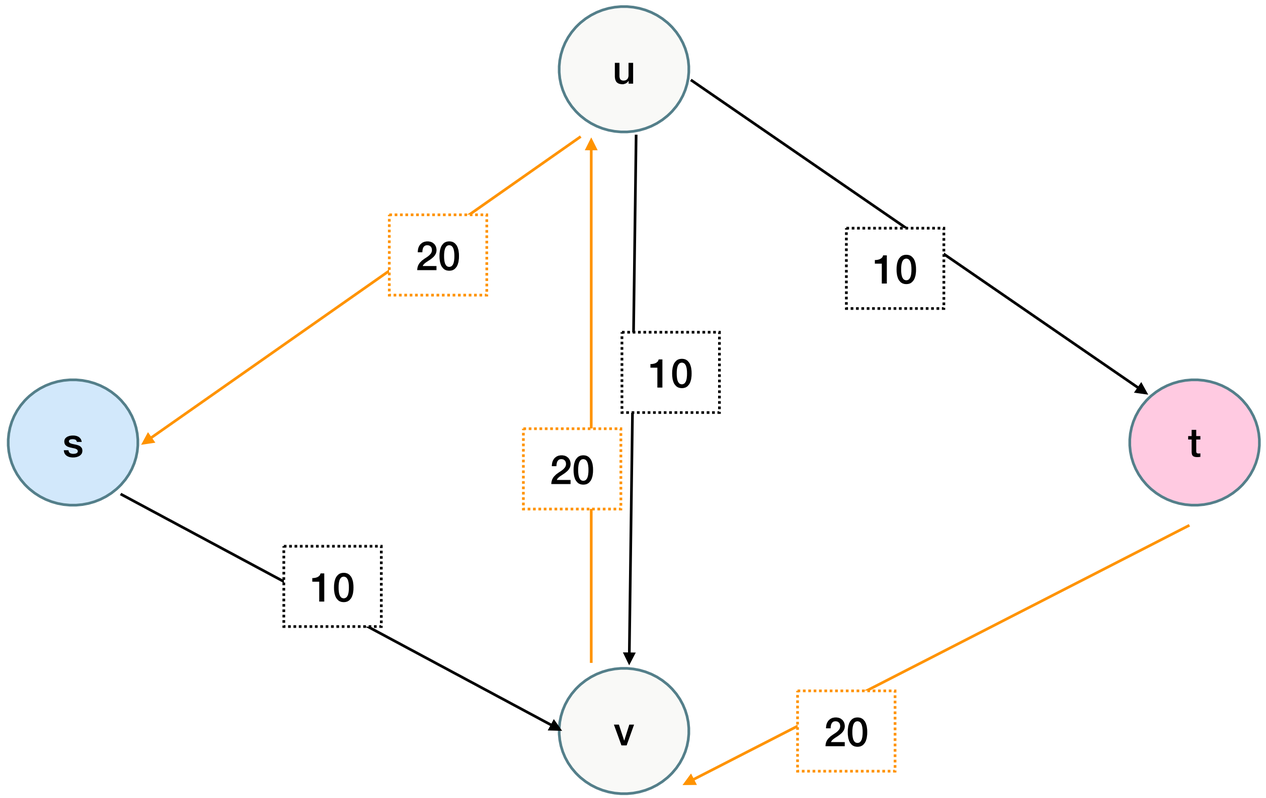
```
MaxFlow (G)
  set f(e) = 0 for all e in G
  while there is an s-t path in Gf
    let P be a simple s-t path in Gf
    f' = augment(f, P)
    f = f'
    Gf = Gf'
  endwhile
 return f
```

Recall: Basic Algorithm



Initially, flow is 0, and $G_f = G$

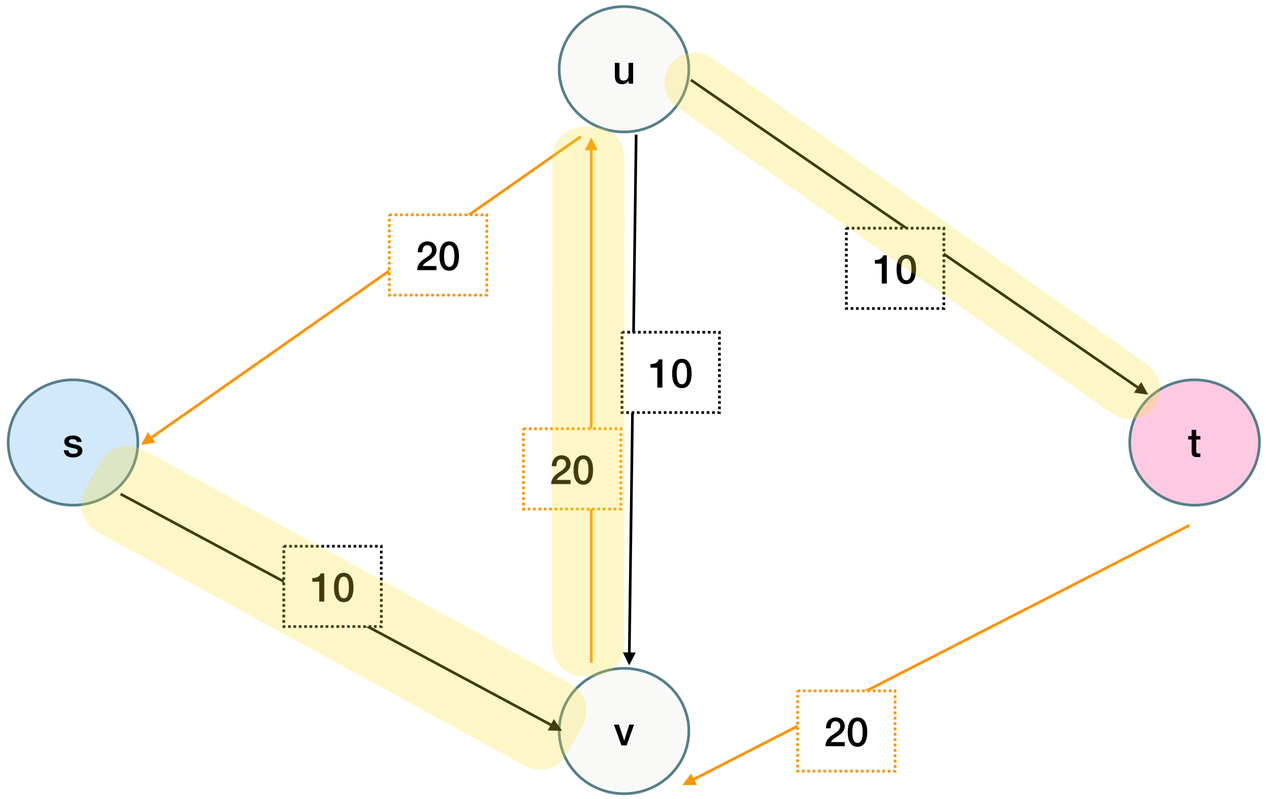
Recall: Basic Algorithm



S -> u -> v -> t is chosen, and we push 20 units across it

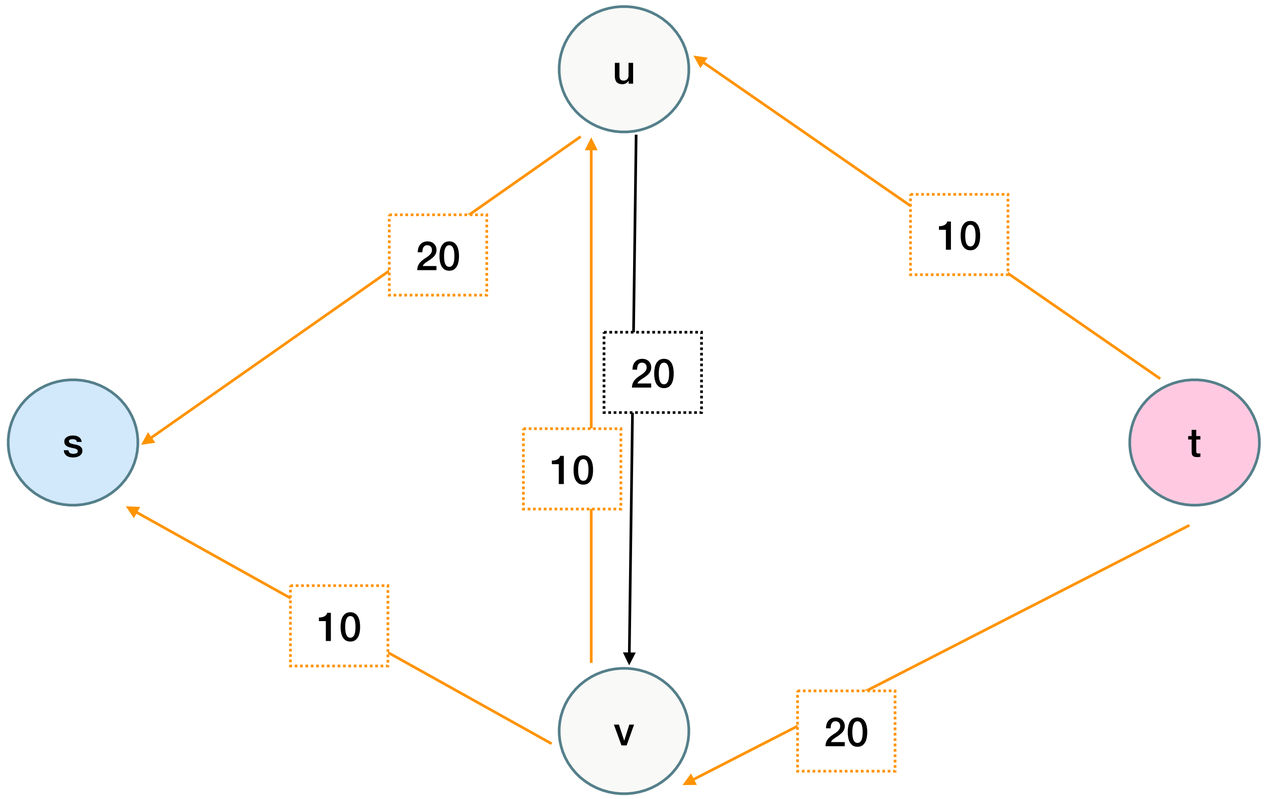
We update G_f

Recall: Basic Algorithm



S -> v -> u -> t is chosen, and we push 10 units across it

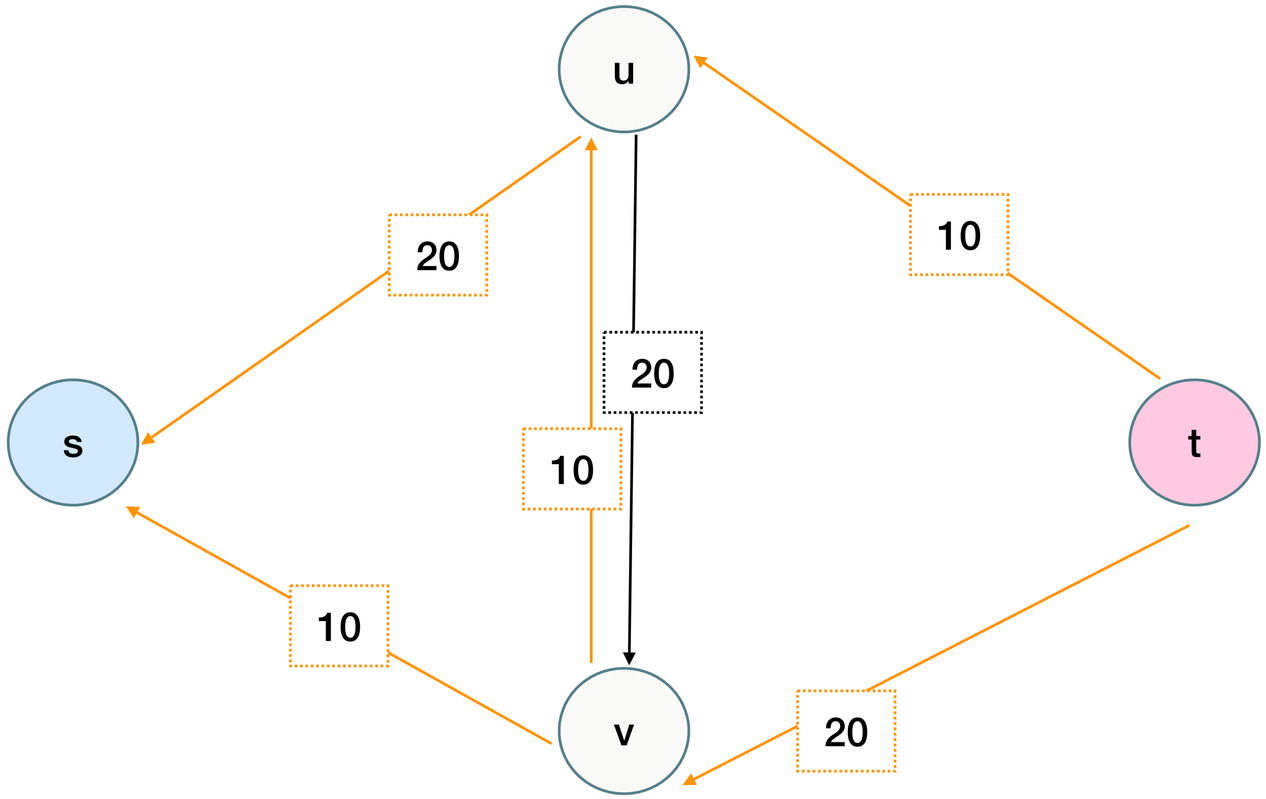
Recall: Basic Algorithm



S -> v -> u -> t is chosen, and we push 10 units across it

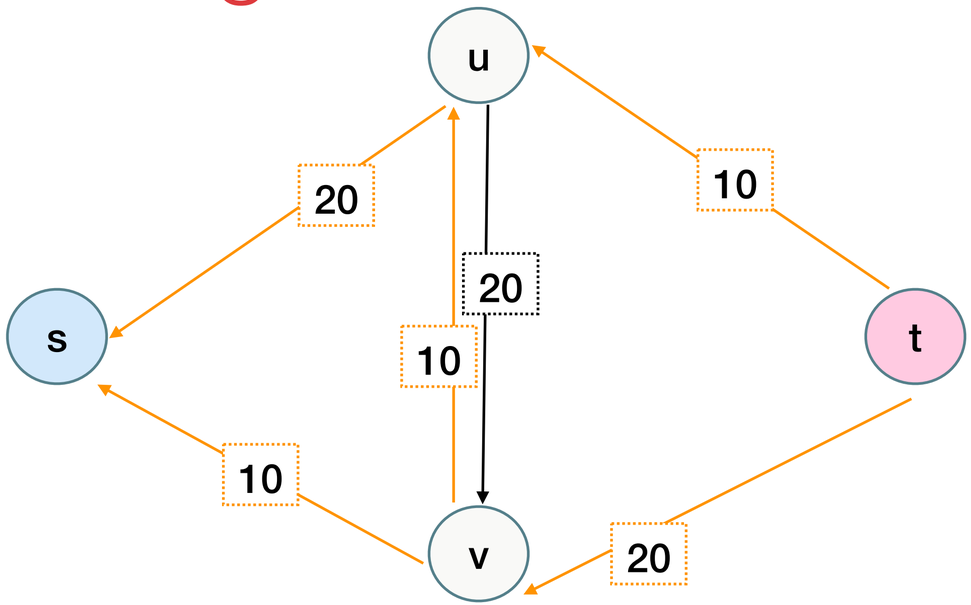
We update G_f

Recall: Basic Algorithm



No more s-t paths in the residual graph. Algorithm terminates with v(f) = 30 (the maximum).

Recall: Basic Algorithm



This is the basic Ford-Fulkerson algorithm. Running time is O(mC)

Strongly-polynomial algorithms exist (e.g. Dinitz-Edmonds-Karp $O(nm^2)$)

One can reduce Max Flow with minimum bounds to Max Flow

One can also add gain / loss as is necessary in StringTie

Bias-aware MaxFlow algorithm

Max-flow procedure

Input: flow network with sink, source, multipliers (b_v as defined in Methods), and capacities on the edges

Output: maximum flow $flow_{max}$

Initialization: set $flow_{max}=0$
for all edges (u,v) in network
$flow_{uv}=0$
end for

while there is an augmenting path p^2 in network
set u to *sink* (the last node in p)
$bias(u) = 1$
$increment = \infty$
while there is predecessor v of node u in p
$increment = \min(increment, capacity_{vu} - bias(u) * flow_{vu})$
if there is a predecessor w of v in p
if v comes before u in the ASG
if w comes before v in the ASG
$bias(v)=bias(u)*b_v$
else $bias(v)=bias(u)$
end if
else if w comes before v in the ASG
$bias(v)=bias(u)$
else $bias(v)=bias(u)/b_v$
end if
end if
end if
$u = v$
end while

for all consecutive nodes u,v (u before v) in p
$flow_{uv}$ += $increment/bias(u)$
$flow_{vu}$ -= $increment/bias(u)$
end for

$flow_{max}$ += $increment$

end while

Processing the Splice Graph

Repeat:

1. Heuristically choose a “heavy” path (a path with the heaviest node) in the ASG

2. Estimate path expression by computing max-flow in a flow graph corresponding to this sub-path of the ASG. Subtract the read mass assigned to the nodes in this path & repeat.

Until:

Coverage of heaviest path falls below 2.5 reads per-base

Interestingly: Unlike other approaches that try to use the flow graph to ***find*** and ***quantify*** the paths, StringTie uses a heuristic to select the transcript, and flow only to quantify the selected path.

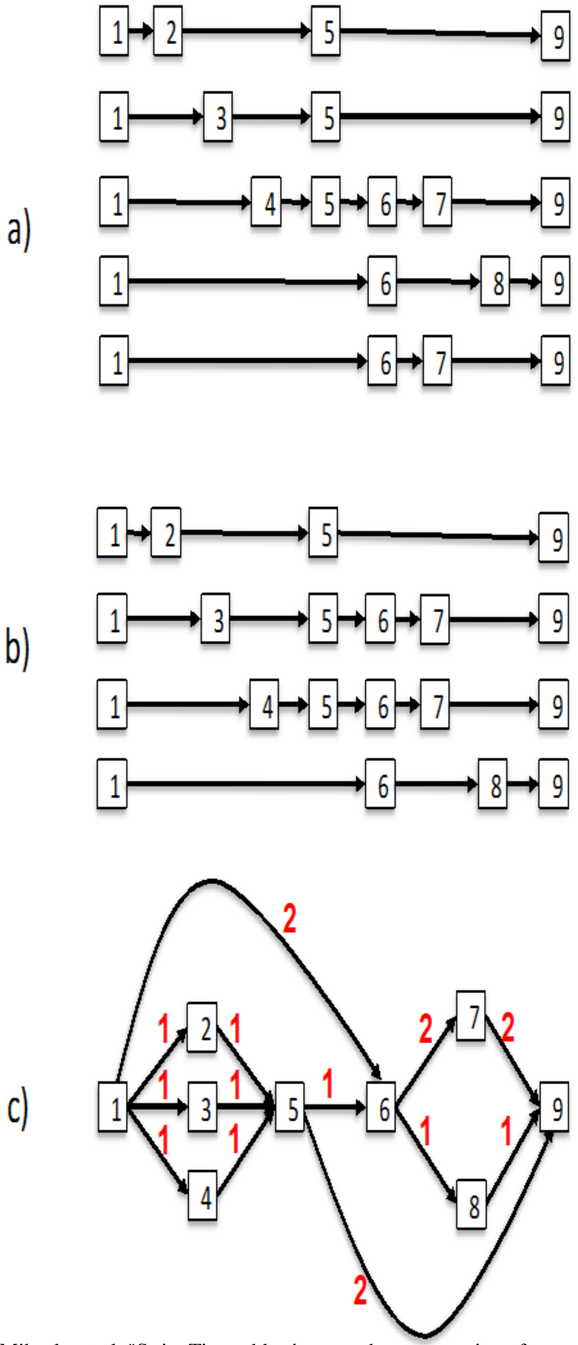


Illustration of how coverage information yields different results from Cufflinks' "parsimony" approach

transcripts extracted if we consider coverage

parsimonious set of "covering" transcripts

ASG

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

Results

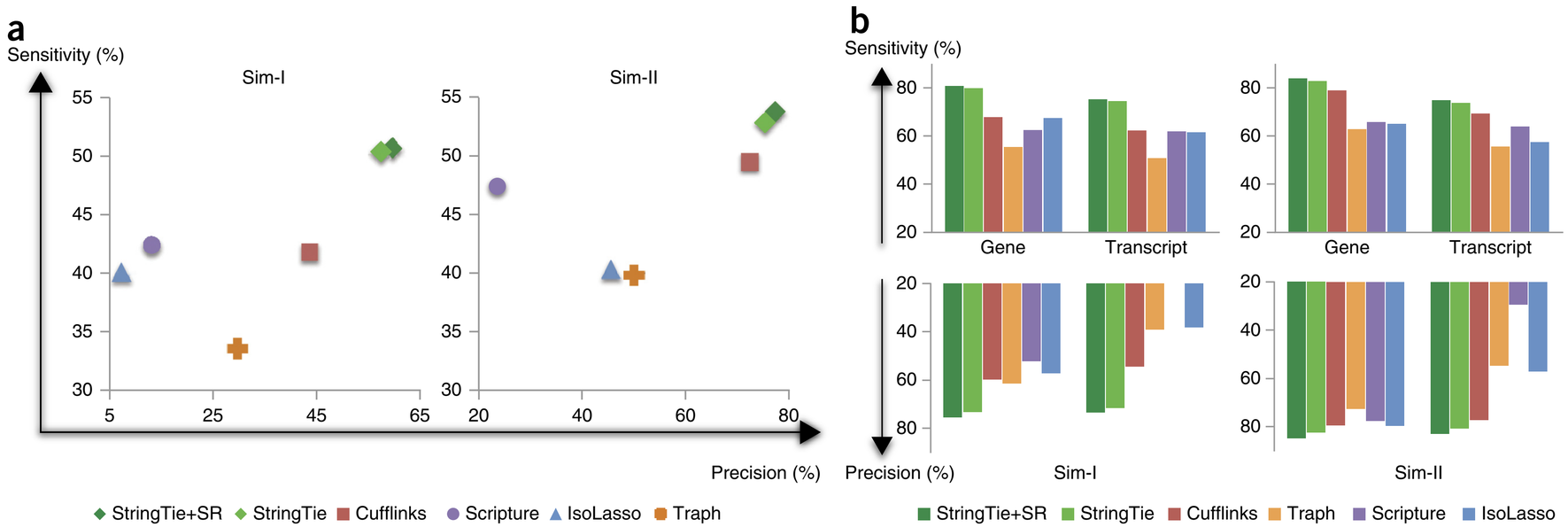


Figure 2 Transcriptome assemblers' accuracies in detecting expressed transcripts from two simulated RNA-seq data sets. (**a**) Transcriptome assemblers' accuracies in detecting expressed transcripts from two simulated RNA-seq data sets. In data set Sim-I (left), the fragment sizes follow an empirical distribution based on Illumina sequences, and in Sim-II (right) the fragment sizes follow a parameterized normal distribution. StringTie+SR pre-assembles the reads into super-reads when possible. (**b**) Accuracy of transcriptome assemblers on gene loci from the same two data sets, considering only those transcripts that were completely covered by input reads. Scripture's precision on Sim-I was 17.7%, below the 20% minimum shown here.

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

Results

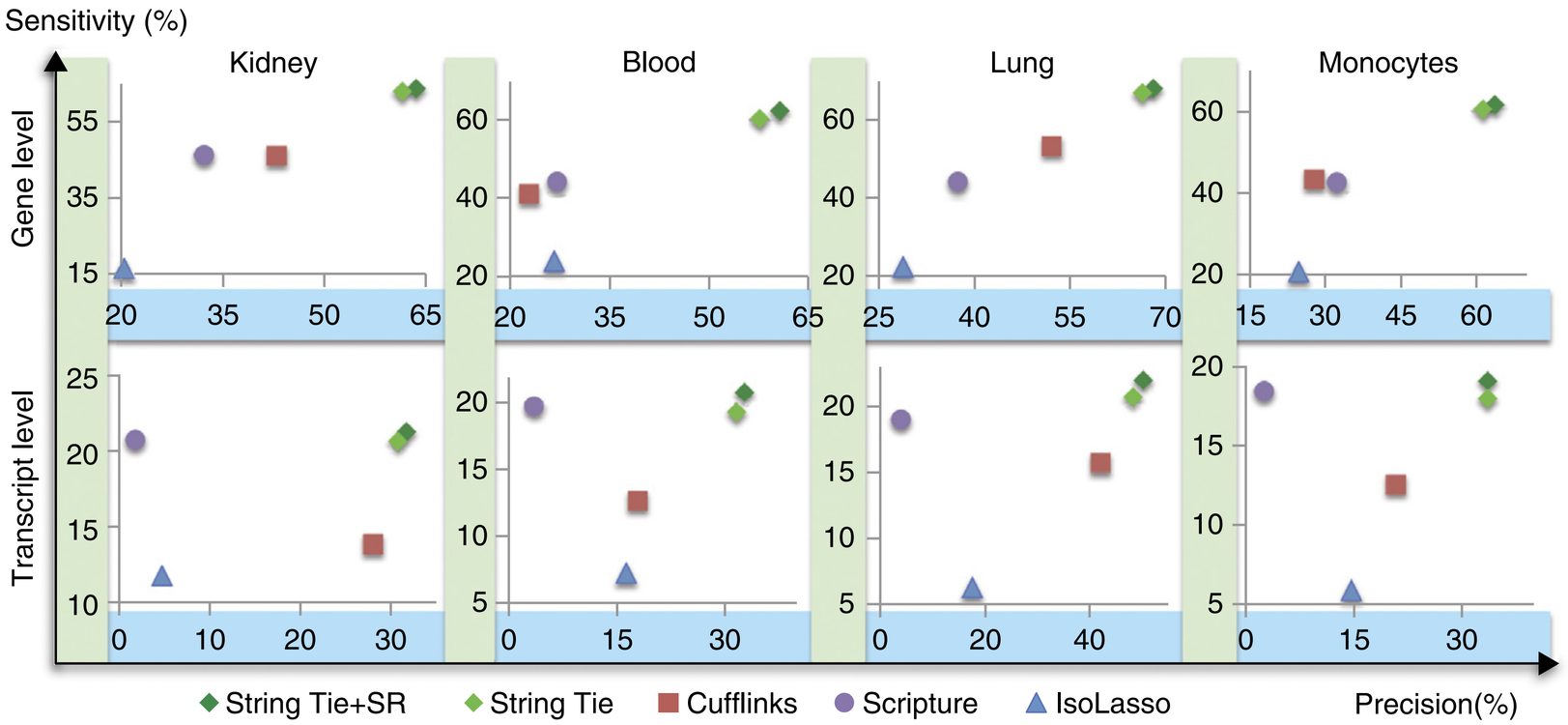


Figure 3 Accuracy of transcript assemblers at assembling known genes, measured on real data sets from four different tissues. Known genes are defined as those annotated in either the RefSeq, UCSC or Ensembl human gene databases. Gene level sensitivity (*y* axis) measures the percentage of genes for which a program got at least one isoform correct, whereas transcript sensitivity measures the percentage of known transcripts that were correctly assembled. Precision (*x* axis) is measured as the percentage of all predicted genes (transcripts) that match an annotated gene (transcript).

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

StringTie Identifies More Transcripts Than Cufflinks

Supplementary Table 11. Symmetric differences beween StringTie and Cufflinks on four real data sets. For each data set, the table shows the number of transcripts identified correctly by StringTie but missed by Cufflinks, the number identified by Cufflinks but missed by StringTie, and the number identified correctly by both programs.

Data set	Unique to StringTie	Unique to Cufflinks	Common to both
Kidney	6652	2177	7068
Blood	5834	2031	5156
Lung	5272	1937	8434
Monocytes	5296	2050	5452

Pertea, Mihaela, et al. "StringTie enables improved reconstruction of a transcriptome from RNA-seq reads." *Nature biotechnology* 33.3 (2015): 290.

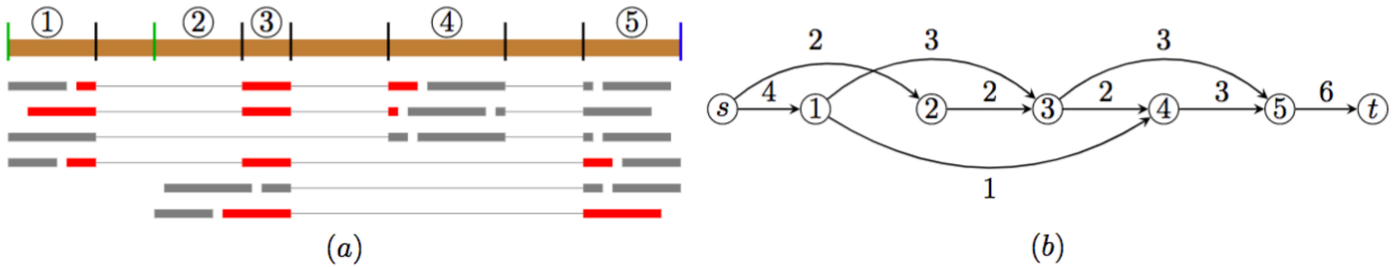
Scallop improves assembly by preserving “phasing paths”

BRIEF COMMUNICATIONS

nature biotechnology

Accurate assembly of transcripts through phase-preserving graph decomposition

Mingfu Shao & Carl Kingsford



Supplementary Figure 1: Example of building splice graph and phasing paths. **(a)** Alignment of reads to the reference genome. Inferred splice positions are marked with black bars; inferred starting and ending positions are marked with green and blue bars, respectively. Exons and partial exons are labeled with numbers above the reference genome. Reads that span more than two exons are colored red, from which we can get the set of phasing paths as $\{(1,3,4),(2,3,5),(1,3,5)\}$. The abundance of these phasing paths are $g(1,3,4)=2$, $g(2,3,5)=1$, and $g(1,3,5)=1$. **(b)** The corresponding splice graph and weights for all edges.

phasing paths = sub-paths of the splicing graph where read evidence supports >1 splicing junction.

Scallop preserves observed sub-paths.

Shao, Mingfu, and Carl Kingsford. "Accurate assembly of transcripts through phase-preserving graph decomposition." *Nature biotechnology* 35.12 (2017): 1167.

Preserving “phasing paths” improves accuracy

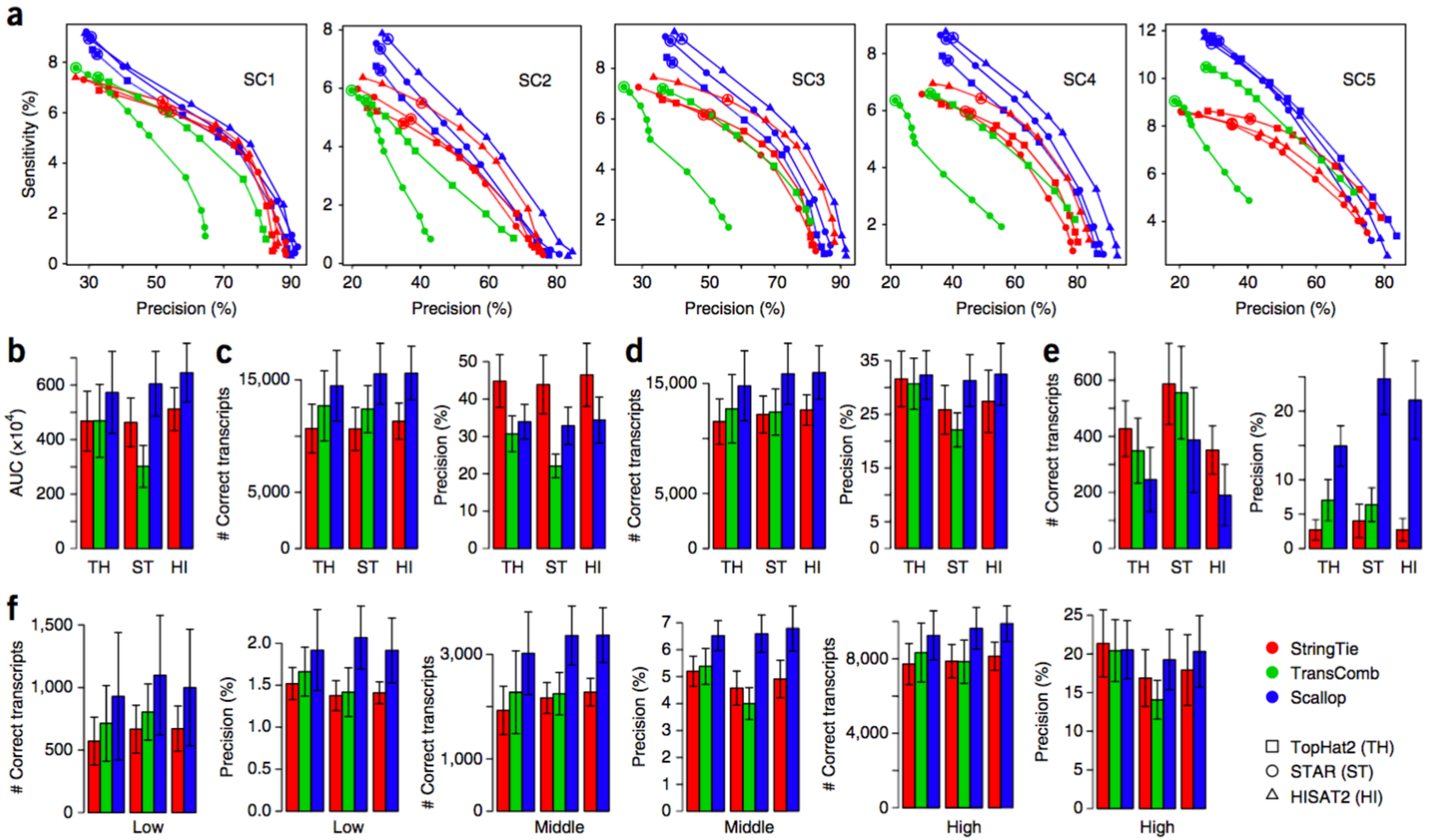


Figure 1 Comparison of the three methods (StringTie, TransComb, and Scallop) over the five testing samples. (**a**) The precision-sensitivity curves for multi-exon transcripts. Each curve connects ten points, corresponding to minimum coverage thresholds {0, 1, 2.5, 5, 7.5, 10, 25, 50, 75, 100}; the default value is circled. (**b**) The average AUC (area under the precision-sensitivity curve) over the five samples. The error bars show the s.d. (the same for other panels). (**c**) The average sensitivity and precision of multi-exon transcripts running at default parameters. (**d**) The average sensitivity and precision of multi-exon transcripts running with minimum coverage set to 0. (**e**) The average sensitivity and precision of single-exon transcripts running at default parameters. (**f**) The average sensitivity and precision of multi-exon transcripts corresponding to low, middle, and high expression levels running with minimum coverage set to 0.

Shao, Mingfu, and Carl Kingsford. "Accurate assembly of transcripts through phase-preserving graph decomposition." *Nature biotechnology* 35.12 (2017): 1167.

Some other interesting assemblers

RESEARCH ARTICLE

Strawberry: Fast and accurate genome-guided transcript reconstruction and quantification from RNA-Seq

Ruolin Liu, Julie Dickerson*

METHOD Open Access

Bayesian transcriptome assembly

Lasse Maretty†, Jonas Andreas Sibbesen† and Anders Krogh*

Gene expression

Sparselso: a novel Bayesian approach to identify alternatively spliced isoforms from RNA-seq data

Xu Shi[1], Xiao Wang[1], Tian-Li Wang[2], Leena Hilakivi-Clarke[3], Robert Clarke[3] and Jianhua Xuan[1,*]

These methods, in particular, solve identification and quantification simultaneously.

Conceptually, this seems like the strongest approach
given how related the problems are.