Supplementary Material for "Cov-trans: An Efficient Algorithm for Discontinuous Transcript Assembly in Coronaviruses"

Xiaoyu Guo[1], Zhenming Wu[1], Shu Zhang[1], and Jin Zhao[1,*]

[1]Dept. School of Computer Science and Technology, Qingdao University, Ningxia Road, 266301, Qingdao, Shandong Province, China

[*]Correspondence: zhaojin@qdu.edu.cn

Contents:

1. Detailed evaluation results

1.1. Comparison of Assembly Performance on Simulated Datasets

The detailed performance of the assemblers on all simulated datasets is shown in Tables S1 – S3.

Table S1 Benchmarking results for the datasets of SARS-CoV-2.

Dataset	Parameters	StringTie	Scallop	Jumper	Cov-trans
Sars2-Sim1	F_1-score	0.100	0.091	0.100	**0.667**
	recall	0.053	0.053	0.053	**0.526**
	precision	**1.000**	0.333	**1.000**	0.909
Sars2-Sim2	F_1-score	0.100	0.174	0.095	**0.750**
	recall	0.053	0.105	0.053	**0.632**
	precision	**1.000**	0.500	0.500	0.923
Sars2-Sim3	F_1-score	0.100	0.296	0.174	**0.750**
	recall	0.053	0.211	0.105	**0.632**
	precision	**1.000**	0.500	0.500	0.923
Sars2-Sim4	F_1-score	0.100	0.174	0.400	**0.750**
	recall	0.053	0.105	0.316	**0.623**
	precision	**1.000**	0.500	0.545	0.923
Sars2-Sim5	F_1-score	0.100	0.250	0.294	**0.727**
	recall	0.053	0.158	0.263	**0.632**
	precision	**1.000**	0.600	0.333	0.857
Sars2-Sim6	F_1-score	0.100	0.182	0.323	**0.788**
	recall	0.053	0.105	0.263	**0.684**
	precision	**1.000**	0.667	0.417	0.929

Table S2 Benchmarking results for the datasets of SARS-CoV-1.

Dataset	Parameters	StringTie	Scallop	Jumper	Cov-trans
Sars1-Sim1	F_1-score	0.105	0.087	0.111	**0.813**
	recall	0.059	0.059	0.059	**0.765**
	precision	0.500	0.167	**1.000**	0.867
Sars1-Sim2	F_1-score	0.111	0.273	0.276	**0.686**
	recall	0.059	0.176	0.176	**0.706**
	precision	**1.000**	0.600	0.600	0.667
Sars1-Sim3	F_1-score	0.105	0.364	0.593	**0.848**
	recall	0.059	0.235	0.471	**0.824**
	precision	0.500	0.800	0.800	**0.875**
Sars1-Sim4	F_1-score	0.105	0.480	0.387	**0.788**
	recall	0.059	0.353	0.353	**0.765**
	precision	0.500	0.750	0.429	**0.813**
Sars1-Sim5	F_1-score	0.111	0.250	0.387	**0.765**
	recall	0.059	0.176	0.353	**0.765**
	precision	**1.000**	0.429	0.429	0.765
Sars1-Sim6	F_1-score	0.111	0.480	0.345	**0.875**
	recall	0.059	0.353	0.294	**0.824**
	precision	**1.000**	0.750	0.417	0.933

Table S3 Benchmarking results for the datasets of MERS-CoV.

Dataset	Parameters	StringTie	Scallop	Jumper	Cov-trans
Mers-Sim1	F_1-score	0.074	0.378	0.077	**0.681**
	recall	0.040	0.280	0.040	**0.640**
	precision	0.500	0.583	**1.000**	0.727
Mers-Sim2	F_1-score	0.077	0.595	0.071	**0.708**
	recall	0.040	0.440	0.040	**0.680**
	precision	**1.000**	0.917	0.333	0.739
Mers-Sim3	F_1-score	0.077	0.571	0.200	**0.708**
	recall	0.040	0.480	0.120	**0.680**
	precision	**1.000**	0.706	0.600	0.739
Mers-Sim4	F_1-score	0.077	0.696	0.500	**0.723**
	recall	0.040	0.640	0.440	**0.680**
	precision	**1.000**	0.762	0.579	0.773
Mers-Sim5	F_1-score	0.077	**0.773**	0.511	0.708
	recall	0.040	**0.680**	0.480	**0.680**
	precision	**1.000**	0.895	0.545	0.739
Mers-Sim6	F_1-score	0.077	**0.727**	0.571	0.708
	recall	0.040	0.640	0.480	**0.680**
	precision	**1.000**	0.842	0.706	0.739

1.2. The performance of Cov-trans under different thresholds

The Depth of a canonical transcripts refers to the average number of times this transcript is sequenced across the dataset. In this study, the depth threshold is defined as the maximum value between 0.001 times the average depth of all vertices in the discontinuous graph. The performance of Cov-trans under different depth thresholds is shown in Table S4.

Table S4 The performance of Cov-trans under different thresholds of depth

Dataset	Depth Thresholds	F1-score	recall	precision
Sars2-sim1	0.0015	0.737	0.583	1.000
	0.0010	0.800	0.667	1.000
	0.0005	0.762	0.667	0.889
Sars1-sim1	0.0015	0.545	0.400	0.857
	0.0010	0.857	0.800	0.923
	0.0005	0.774	0.800	0.750
Mers-sim1	0.0015	0.800	0.727	0.889
	0.0010	0.800	0.727	0.889
	0.0005	0.800	0.727	0.889

1.3. Parameter setting for Mixed-Integer Linear Programming

The specific values of the parameters (w_1 , w_2 , and w_3) used in the mixed-integer linear programming model are provided in Table S5 for simulated datasets and Table S6 for real datasets.

Table S5 The parameters (w_1, w_2, and w_3) of simulated datasets

Dataset	w_1	w_2	w_3	Types of viruses
Sars2-sim1	0.056	0.750	0.514	SARS-CoV-2
Sars2-sim2	0.056	0.692	0.474	
Sars2-sim3	0.500	0.074	0.042	
Sars2-sim4	0.083	0.444	0.250	
Sars2-sim5	0.063	0.533	0.364	
Sars2-sim6	0.091	0.407	0.229	
Sars1-sim1	0.500	0.069	0.048	SARS-CoV-1
Sars1-sim2	0.200	0.156	0.104	
Sars1-sim3	0.500	0.065	0.034	
Sars1-sim4	0.250	0.133	0.071	
Sars1-sim5	0.333	0.094	0.064	
Sars1-sim6	0.500	0.031	0.167	
Mers-sim1	0.067	0.405	0.263	MERS-CoV
Mers-sim2	0.063	0.400	0.258	
Mers-sim3	0.056	0.486	0.277	
Mers-sim4	0.047	0.538	0.350	
Mers-sim5	0.077	0.342	0.224	
Mers-sim6	0.045	0.564	0.319	

Table S6 The parameters (w_1, w_2, and w_3) of real datasets

Dataset	w_1	w_2	w_3	Types of viruses
SRR12789544	0.059	0.436	0.227	SARS-CoV-2
SRR12789545	0.083	0.444	0.250	
SRR12789546	0.667	0.385	0.208	
SRR1942954	0.016	0.685	0.349	SARS-CoV-1
SRR1942956	0.016	0.687	0.354	
SRR1942957	0.016	0.685	0.349	
SRR10357372	0.130	1.000	0.517	MERS-CoV
SRR10357373	0.150	1.046	0.540	
SRR10357374	0.025	0.580	0.300	

2. Command Lines used for benchmarking

For benchmarking, we used three existing tools, which are listed below. All tools were tested with their default settings unless stated otherwise.

- **Jumper arguments**

We run Jumper with the following arguments:

```
python jumper_main.py -b ${input_bam} -f ${input_fasta} --outputDecomposition ${output_decomposition} --outputGTF ${output_assembled} –-sj_threshold {SJ_threshold}
```

- **Scallop arguments**

We run Scallop v0.10.5 with the following arguments:

```
scallop -i ${input_bam} -o ${output_assembled}
```

- **StringTie arguments**

We run StringTie v2.2.1 with the following arguments:

```
stringtie -o ${output_assembled} ${input_bam}
```