bioRxiv preprint doi: https://doi.org/10.1101/283101; this version posted March 16, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-ND 4.0 International license.

Genome Analysis

Tychus: a whole genome sequencing pipeline for assembly, annotation and phylogenetics of bacterial genomes

Christopher Dean[1,†], Noelle Noyes[1,†], Steven Lakin[1,†], Pablo Rovira-Sanz[3], Xiang Yang[4], Keith Belk[3], Paul S. Morley[2], Rick Meinersmann[5], Zaid Abdo[1,*]

[1]Department of Microbiology, Immunology and Pathology, [2]Department of Clinical Sciences, [3]Department of Animal Sciences, Colorado State University, Fort Collins, CO, 80523, USA

[4]Department of Animal Science, University of California, Davis, CA, 95616, USA, and

[5]Bacterial Epidemiology and Antimicrobial Resistance Research, USDA-Agricultural Research Service, Athens, Georgia, USA

Abstract

Summary: Tychus is a tool that allows researchers to perform massively parallel whole genome sequence (WGS) analysis with the goal of producing a high confidence and comprehensive description of the bacterial genome. Key features of the Tychus pipeline include the assembly, annotation, alignment, variant discovery and phylogenetic inference of large numbers of WGS isolates in parallel using open-source bioinformatics tools and virtualization technology. All prerequisite tools and dependencies come packaged together in a single suite that can be easily downloaded and installed on Linux and Mac operating systems.

Availability: Tychus is freely available as an open-source package under the MIT license, and can be downloaded via GitHub (https://github.com/Abdo-Lab/Tychus).

Contact: zaid.abdo@colostate.edu

1 Introduction

The zeitgeist of the genomics era has been defined by the accessibility and affordability of next-generation sequencing platforms, which are capable of producing large amounts of data for genomics research. Whole genome sequencing (WGS) allows for the interrogation and analysis of complete genomes and can be applied to a wide range of organisms including plants, animals, bacteria, and viruses. Recently, WGS methods have been used in the domain of public health in order to better understand and track foodborne illnesses caused by bacterial pathogens. These efforts have led to the development of open-source reference databases and foodborne outbreak detection and investigation initiatives (Barkley et al., 2016; see also Hu et al., 2016). However, the accuracy of WGS for bacterial pathogen identification and outbreak analysis relies not only on the methods used to isolate, extract, and sequence the bacterial DNA, but also on the bioinformatics and statistical analyses applied to the resulting sequence data. In addition, as WGS datasets expand exponentially, there is a critical need for widely-accessible and computationally-efficient analytic pipelines. To this end, we present Tychus, an open-source, easy-install, user-friendly and rapid software tool for performing large-scale, parallel bacterial sequence analysis that produces a high confidence and comprehensive description of the bacterial genome.

* To whom correspondence should be addressed.

† The authors wish it to be known that in their opinion, the first three authors should be regarded as joint first authors.

2 Methods and Implementation

Tychus is written using Nextflow (Tommaso *et al*., 2015), a parallel DSL workflow framework and integrated with Docker (Merkel, 2014), a software containerization platform that resolves the installation and configuration issues of the many open-source bioinformatics tools utilized throughout the pipeline. Tychus is split into two complementary modules (see Figure 1): assembly and alignment, as described below.

2.1 Assembly Module

Because no single assembler consistently produces an optimal genome assembly (Bradnam et al., 2013) in isolation, we utilize the results of four *de novo* genome assemblers to construct a consensus assembly of higher contiguity and accuracy. This module also supports methods for identifying and classifying genomic features of interest, a process called annotation, as well as reporting on common quality score metrics for each resulting assembly.

Input to the Tychus assembly module consists of a collection of paired fastq files. These files are preprocessed using Trimmomatic (Bolger, Lohse, & Usadel, 2014) to remove adapter sequences, low quality base pairs and sequence fragments. This prevents adapter sequences from being used to build the resulting assemblies, and has been shown to improve the quality of *de novo* assemblies as measured by their N50 statistic (2014). Preprocessed reads are then assembled with Abyss

bioRxiv preprint doi: https://doi.org/10.1101/283101; this version posted March 16, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-ND 4.0 International license.

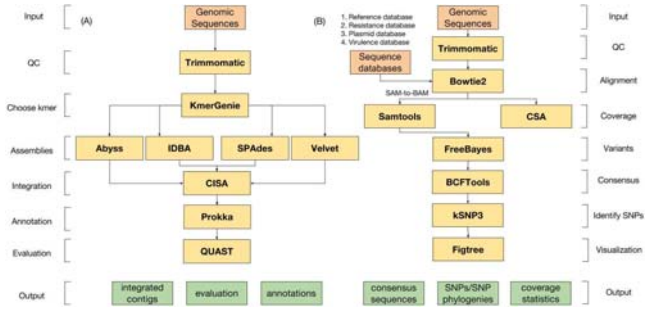

Fig. 1. Implementation diagram for each step in the assembly and alignment modules (inputs, tools utilized, and outputs).

(Simpson et al., 2009), IDBA-UD (Peng et al., 2012), SPAdes (Bankevich et al., 2012), and Velvet (Zerbino & Birney, 2008). Prior to *de novo* assembly, an important step is to choose a *k*-mer value with which to build the underlying De Bruijn graph. As Abyss and Velvet do not iterate on multiple *k*-mer values, KmerGenie (Chikhi & Medvedev, 2014) is used to optimize this value prior to assembly. Contigs produced from each assembler are combined and integrated using CISA (Lin & Liao, 2013) to construct a super (consensus) assembly of higher quality and contiguity. Prokka (Seemann, 2014) is then used to annotate the super assembly. Lastly, contigs are evaluated with QUAST (Gurevich et al., 2013) to evaluate various common assembly score metrics, such as number of contigs, largest contig, and N50.

2.2 Alignment Module

The Tychus alignment module takes in six inputs: a collection of paired-end fastq files, a fasta-formatted reference genome, and optional databases of plasmid, resistance, virulence factor, and draft genome nucleotide sequences. Similar to the assembly module, reads are first quality-filtered using Trimmomatic (Bolger, Lohse, & Usadel, 2014), which helps to decrease the number of misalignments downstream. Next, Bowtie2 (Langmead & Salzberg, 2012) is used to align reads to the user-input reference genome, as well as the plasmid, resistance, and virulence databases. The results from alignment to the resistance, virulence and plasmid databases are used by an in-house tool to determine the overall coverage of each feature (plasmids, virulence factors, and antimicrobial resistance genes) present in each sample, with higher coverage features reducing the number of false-positive gene identifications. Files resulting from alignment to the reference genome are utilized by Freebayes (Garrison & Marth, 2012) to identify sequence variants and single nucleotide polymorphisms (SNPs), which are utilized by BCFtools (Li, 2011) to obtain consensus sequences for each sample. These consensus sequences, in addition to the optional genomes produced from the assembly module, act as draft genomes, which can be used by kSNP3 (Gardner, Slezak, & Hall, 2015) to identify related SNPs and build SNP phylogenies. The Newick formatted phylogenies produced by kSNP3 (2015) are then rendered into a user-defined image format using the command-line version of Figtree (https://github.com/rambaut/figtree).

3 Case Study

We used 15 *L. monocytogenes* samples (acc. no. PRJNA374745) with a reference genome (acc. no. PRJNA61583) and a 64-core (AMD Opteron Processor 6378) Ubuntu server to provide results for a common use case

Table 1. Assembly results for 15 *L. monocytogenes* samples.

		S036	S037	S038	S040	S041	S042	S043	S044	S056	S093	S095	S097	S099	S101	S103
Abyss	# contigs	59	120	63	27	31	70	35	127	19	32	41	31	47	14	15
	Largest contig	335,980	207,467	329,432	550,131	748,468	294,458	378,754	159,856	595,207	408,585	344,175	824,747	358,382	1,257,175	1,180,011
	N50	109,496	39,120	127,399	312,765	247,200	65,681	207,493	39,892	390,544	203,305	207,504	350,896	132,132	543,925	421,870
IDBA	# contigs	20	50	38	24	24	29	32	40	20	24	25	31	24	17	24
	Largest contig	608,550	434,230	546,931	891,939	891,939	660,784	434,898	433,202	1,243,429	779,103	542,550	891,919	745,546	1,256,470	895,457
	N50	275,337	157,464	327,831	588,700	588,700	332,478	333,394	155,989	577,456	297,832	323,661	346,158	438,484	524,509	291,726
SPAdes	# contigs	11	37	19	10	10	16	18	26	10	13	13	14	16	7	12
	Largest contig	609,944	363,494	571,501	1,221,296	1,221,050	579,208	1,025,352	424,099	1,245,082	1,045,660	711,786	1,221,268	745,926	1,512,478	1,182,006
	N50	501,135	178,796	503,215	588,659	588,659	491,509	434,949	227,398	578,264	438,946	505,806	546,879	517,517	1,512,478	433,615
Velvet	# contigs	66	96	43	22	22	40	21	58	16	34	37	22	29	11	16
	Largest contig	177,730	164,049	343,809	501,114	492,848	346,140	486,043	213,403	854,885	344,884	348,896	553,064	349,083	1,511,967	1,181,698
	N50	90,623	55,410	152,246	273,753	200,526	173,915	375,839	88,991	689,565	174,436	181,109	302,499	179,220	1,511,967	460,697
CISA	# contigs	11	36	8	10	9	14	11	23	11	12	14	8	13	7	9
	Largest contig	609,964	435,935	650,166	1,222,947	1,222,701	663,416	1,026,165	434,947	1,246,139	1,046,362	713,473	1,351,366	745,970	1,512,478	1,182,028
	N50	502,833	180,470	546,931	590,405	590,405	491,509	436,695	271,699	689,596	439,990	506,398	572,601	519,241	1,512,478	460,726

for each Tychus module. By default, all 15 samples were run in parallel using all available computing cores.

The assembly module was run with default parameters for each sample. Assemblies, annotations, and all assembly summary statistics were computed in under an hour, using approximately 40 gigabytes (GB) of RAM. Consistent with previous findings (Lin & Liao, 2013) the CISA integrated contigs produced superior assemblies in terms of N50, contig length, and number of contigs in nearly all samples (see Table 1).

Similarly, the alignment module was run with default settings for each sample. All sequence alignments, consensus sequences, SNPs, SNP phylogenies, and phylogenetic trees were produced in under 30 minutes using approximately 18 GB of RAM. A total of 77,601 SNPs were identified from all 15 strains, with 36,503 of these identified as core (SNPs present in all genomes); 41,098 identified as non-core; and 56,894 identified as majority SNPs present in a user-defined fraction of all samples (default 0.75).

In the end, it took Tychus <2 hours and <41GB of RAM to turn raw WGS data for 15 samples into high-quality assemblies with complete annotation and robust phylogenetic trees.

Conclusion

Tychus is built upon existing open-source virtualization technology and provides a pipeline framework that is intuitive and easy-to-use for both novices and developers. It takes advantage of emerging multi-core computing architectures and large memory servers to deliver results quickly and reliably without the need for extensive bioinformatics or computing expertise. Though Tychus can be used on modest machines (with a penalty to run time), it is intended for use on servers with large amounts of RAM and disk space and with multiple computing cores.

Acknowledgements

Funding: This work has been supported by startup funding from Colorado State University provided to ZA.

Conflict of Interest: none declared.

References

Barkley, J. S., Gosciminski, M., & Miller, A. (2016). Whole-Genome Sequencing Detection of Ongoing Listeria Contamination at a Restaurant, Rhode Island, USA, 2014. *Emerging Infectious Diseases, 22*(8), 1474–1476. https://doi.org/10.3201/eid2208.151917

Bankevich, A., Nurk, S., Antipov, D., Gurevich, A. A., Dvorkin, M., Kulikov, A. S., … Pevzner, P. A. (2012). SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing. *Journal of Computational Biology, 19*(5), 455–477. https://doi.org/10.1089/cmb.2012.0021

Bradnam, K. R., Fass, J. N., Alexandrov, A., Baranay, P., Bechner, M., Birol, I., … Korf, I. F. (2013). Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species. *GigaScience, 2*(1), 10. https://doi.org/10.1186/2047-217X-2-10

bioRxiv preprint doi: https://doi.org/10.1101/283101; this version posted March 16, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-ND 4.0 International license.

Bolger, A. M., Lohse, M., & Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina Sequence Data. *Bioinformatics*, btu170. https://doi.org/10.1093/bioinformatics/btu170

Garrison, E., & Marth, G. (2012). Haplotype-based variant detection from short-read sequencing. *arXiv:1207.3907 [q-Bio]*. Retrieved from http://arxiv.org/abs/1207.3907

Hu, K., Renly, S., Edlund, S., Davis, M., & Kaufman, J. (2016). A modeling framework to accelerate food-borne outbreak investigations. *Food Control*, *59*, 53–58. https://doi.org/10.1016/j.foodcont.2015.05.017

Li, H. (2011). A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics*, *27*(21), 2987–2993. https://doi.org/10.1093/bioinformatics/btr509

Lin, S.-H., & Liao, Y.-C. (2013). CISA: Contig Integrator for Sequence Assembly of Bacterial Genomes. *PLoS ONE*, *8*(3). https://doi.org/10.1371/journal.pone.0060843

Merkel, D. (2014). Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux J.*, *2014*(239). Retrieved from http://dl.acm.org/citation.cfm?id=2600239.2600241

Peng, Y., Leung, H. C. M., Yiu, S. M., & Chin, F. Y. L. (2012). IDBA-UD: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics (Oxford, England)*, *28*(11), 1420–1428. https://doi.org/10.1093/bioinformatics/bts174

Seemann, T. (2014). Prokka: rapid prokaryotic genome annotation. *Bioinformatics (Oxford, England)*, *30*(14), 2068–2069. https://doi.org/10.1093/bioinformatics/btu153

Simpson, J. T., Wong, K., Jackman, S. D., Schein, J. E., Jones, S. J. M., & Birol, İ. (2009). ABySS: A parallel assembler for short read sequence data. *Genome Research*, *19*(6), 1117–1123. https://doi.org/10.1101/gr.089532.108

Simpson, Jared T., Kim Wong, Shaun D. Jackman, Jacqueline E. Schein, Steven J.M. Jones, and İnanç Birol. "ABySS: A Parallel Assembler for Short Read Sequence Data." *Genome Research* 19, no. 6 (June 2009): 1117–23. doi:10.1101/gr.089532.108.

Tommaso, P. D., Chatzou, M., Prieto, P., Palumbo, E., Notredame, C. (2014). A novel tool for highly scalable computational pipelines. figshare. https://doi.org/https://doi.org/10.6084/m9.figshare.1254958.v2

Zerbino, D. R., & Birney, E. (2008). Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, *18*(5), 821–829. https://doi.org/10.1101/gr.074492.107