A Practical Comparison of *De Novo* Genome Assembly Software Tools for Next-Generation Sequencing Technologies

Wenyu Zhang, Jiajia Chen, Yang Yang, Yifei Tang, Jing Shang, Bairong Shen*

Center for Systems Biology, Soochow University, Suzhou, Jiangsu, China

Abstract

The advent of next-generation sequencing technologies is accompanied with the development of many whole-genome sequence assembly methods and software, especially for *de novo* fragment assembly. Due to the poor knowledge about the applicability and performance of these software tools, choosing a befitting assembler becomes a tough task. Here, we provide the information of adaptivity for each program, then above all, compare the performance of eight distinct tools against eight groups of simulated datasets from Solexa sequencing platform. Considering the computational time, maximum random access memory (RAM) occupancy, assembly accuracy and integrity, our study indicate that string-based assemblers, overlap-layout-consensus (OLC) assemblers are well-suited for very short reads and longer reads of small genomes respectively. For large datasets of more than hundred millions of short reads, *De Bruijn* graph-based assemblers would be more appropriate. In terms of software implementation, string-based assemblers are superior to graph-based ones, of which SOAPdenovo is complex for the creation of configuration file. Our comparison study will assist researchers in selecting a well-suited assembler and offer essential information for the improvement of existing assemblers or the developing of novel assemblers.

Citation: Zhang W, Chen J, Yang Y, Tang Y, Shang J, et al. (2011) A Practical Comparison of *De Novo* Genome Assembly Software Tools for Next-Generation Sequencing Technologies. PLoS ONE 6(3): e17915. doi:10.1371/journal.pone.0017915

Editor: I. King Jordan, Georgia Institute of Technology, United States of America

Received November 20, 2010; **Accepted** February 15, 2011; **Published** March 14, 2011

Funding: This work was supported by grants from the National Nature Science Foundation of China (31040020) and the National 973 Programs of China (2007CB947002, 2010CB945600). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: bairong.shen@suda.edu.cn

Introduction

In recent years, the next-generation sequencing (or deep sequencing) technologies have been evolving rapidly, with the potential to accelerate biological and biomedical research dramatically [1]. However, the downstream analysis of short reads datasets after sequencing is a tough task; one of the biggest challenges for the analysis of high throughput sequencing reads is the whole genome assembly. DNA fragment assembly has a long history since the emergence of the first generation of sequencing technologies [2,3]. The assembly procedure becomes especially difficult when tackling short and high throughput reads with different error profiles [4]. According to the existence of reference information, the assembly procedure can be classified as reference-guide genome assembly and *de novo* genome assembly, of which the former is relatively toilless with the aid of reference genome or proteome information while the later in more challenging. Herein, we focus on the comparison and evaluation of tools for *de novo* assembly of genome sequence.

The genome assemblers generally take a file of short sequence reads and a file of quality-value as the input. Since the quality-value file for the high throughput short reads is usually highly memory-intensive, only a few assemblers, for example, SHARCGS [5], and ALLPATHS-LG [6] adopt it in the posterior assembly process. For the sake of computational memory saving and convenience of data inquiry, high-throughput short reads data is always initially formatted to specific data structure. Currently, existing data structure for this usage can be predominantly classified into two categories: string-based model and graph-based model. We therefore call the corresponding assemblers as string-based and graph-based. String-based assemblers, implemented with Greedy-extension algorithm, are mainly reported for the assembly of small genomes [5,7,8,9], while the latter ones are designed aiming at handling complex genomes [10,11,12].

One of the most intractable bottlenecks for practical assembly of next - generation short reads is how to process repetitive fragments from complicated genomes, especially eukaryote genomes. Intuitively, sequencing with longer reads is a potential solution, while it becomes impractical with limit current of sequencing technology. The paired-end (PE) sequencing can, to some extent, compensate for read length [13]. Several assemblers, such as SSAKE [9], SOAPdenovo [11], AbySS [12], Velvet [14,15], exploit PE sequencing information to reduce gaps from assembled contigs. Another big challenge for the assembly of short reads is the intensive computational time requirement. To decrease the time cost of the assembly procedure, thread parallelization is implemented in a couple of graph-based assemblers [11,12].

At our last enumeration, 24 academic *de novo* genome assemblers, each possessing its own range of application, are developed for short reads datasets from different sequencing platforms in the last few years. These assembly tools with corresponding websites and references are listed in Table S1. We classify and list these assemblers according to their data structure models in Figure 1. In the present study, eight short reads assemblers, representing four various assembly strategies, were benchmarked against two types of simulated short reads datasets derived from four different genomes. Our objective is to gain the assemblers' performance information about computational time and memory cost, assembly accuracy, completeness and size distribution of assembled contigs when each assembler is applied to handle datasets with different data size, then to provide essential information for researchers in choosing suitable tools and for computational biologists to develop novel assemblers. The result indicates that each assembly strategy has its own range of applicability while PE reads and longer reads are indeed with the capability to increase the quality of assembled contigs to some extent, and parallel computing is of great potential in short reads assembly, with which the computational time is notably reduced.

Results

At present, mainly three distinct strategies are applied in short reads assembly. Among them, Greedy-extension is the implementation of string-based method, while *De Bruijn* graph and overlap-layout-consensus (OLC) are two different graph-based approaches. Each assembly tool is suitable for dataset from specific sequencing platform.

For each short reads assembly procedure, less computational time and memory cost is our expectation. The computational time of the assembly process is determined by both the dataset complexity and the assembly strategy. The information about running times, maximum memory occupancies for different assemblers applied to different datasets is illustrated in Figures 2 and 3. For string-based assembler, the time and memory cost is approximately proportionate to dataset size, although it is also affected by the complexity of dataset. Among them, SSAKE runs in rather less time than other peer assemblers, but the RAM usage increases dramatically with augmentation of dataset size. QSRA [7] is developed upon the original VCAKE algorithm, which indeed reduces the computational time, at the cost of RAM occupation. SHARCGS runs in comparable speed as QSRA, however it is highly memory-intensive, even unable to handle *E.coli* short reads dataset with our computer power used in this study. Edena is a typically graph-based assembly tool, which has two running modes: strict and nonstrict modes [16]. For the strict mode, fewer but more accurate contigs are generated, while nonstrict mode acts on the contrary. Compared with string-based tools, Edena is superior in terms of time and RAM utilization. Velvet and SOAPdenovo typify another graph-based method. Similar to Edena, they implement

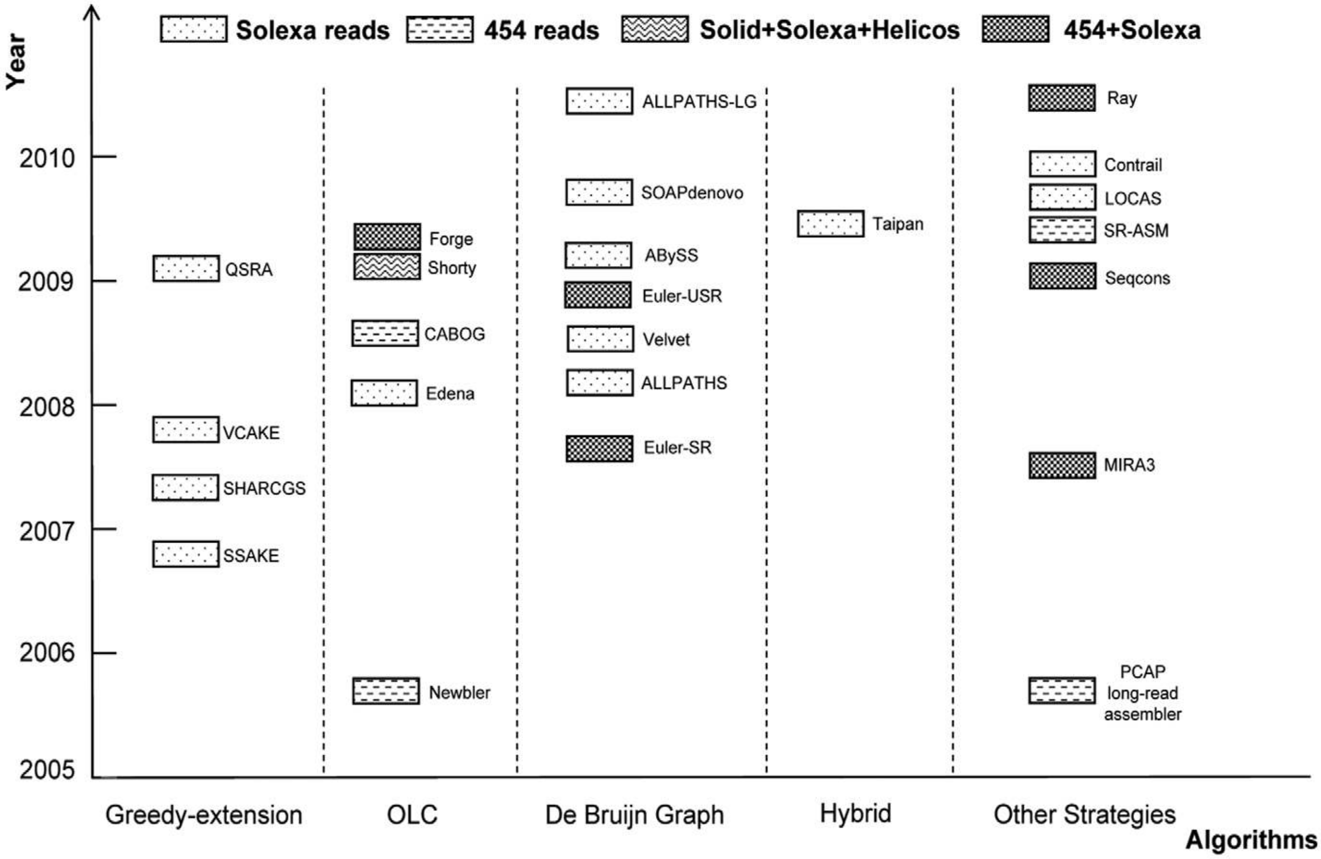


Figure 1. Overview of *de novo* short reads assemblers. Programs developed from year of 2005 to 2010 are classified according to the assembly strategies. Currently, there are mainly four sorts of assemblers, while the other ones are denoted as "Other Strategies". Different box symbols are utilized to distinguish assemblers that for short reads from different platforms.
doi:10.1371/journal.pone.0017915.g001

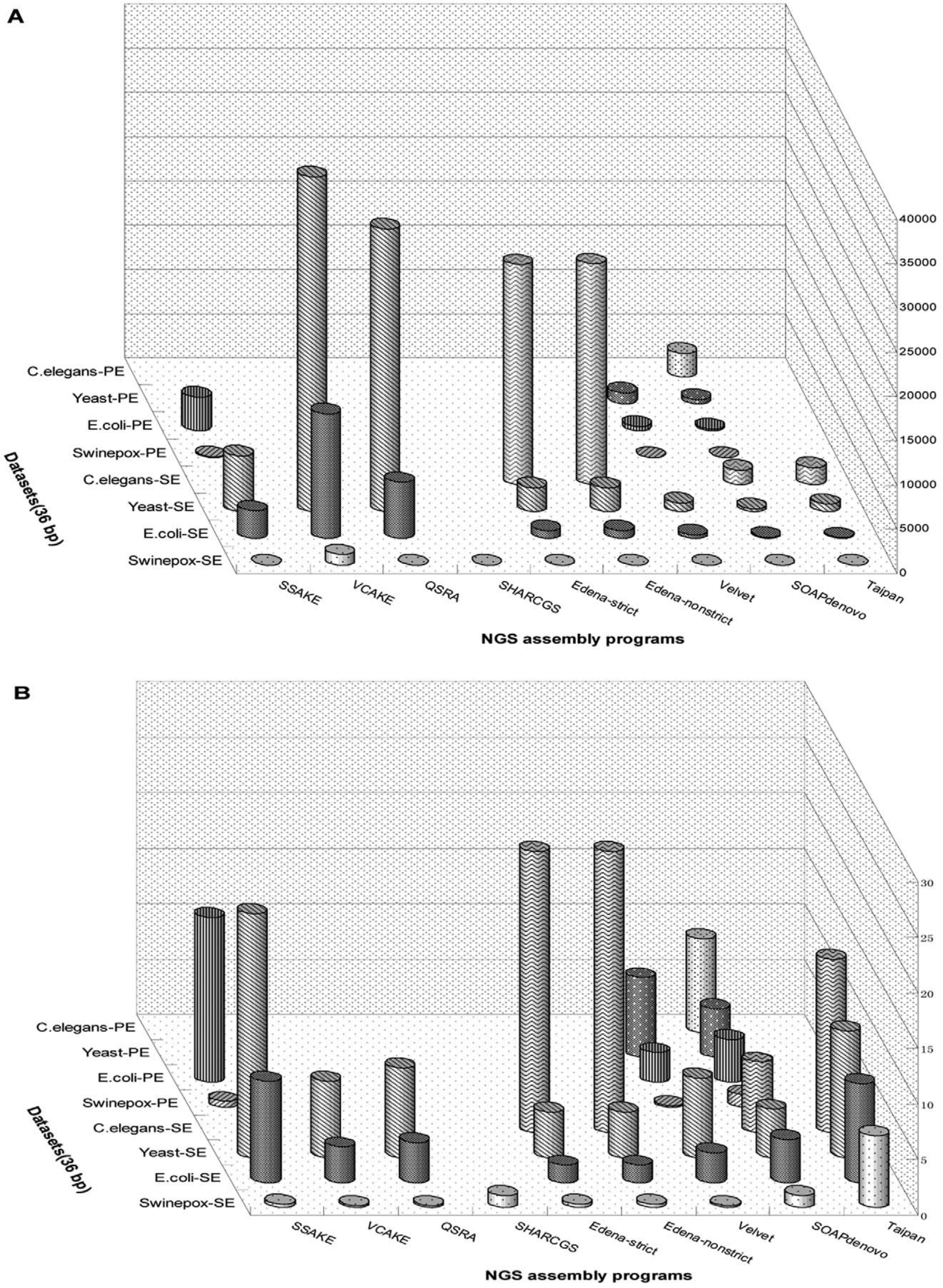

A
C.elegans-PE
Yeast-PE
E.coli-PE
Swinepox-PE
C.elegans-SE
Yeast-SE
E.coli-SE
Swinepox-SE
Datasets(36 bp)
40000
35000
30000
25000
20000
15000
10000
5000
0
Running time(s)
SSAKE
VCAKE
QSRA
SHARCGS
Edena-strict
Edena-nonstrict
Velvet
SOAPdenovo
Taipan
NGS assembly programs
B
C.elegans-PE
Yeast-PE
E.coli-PE
Swinepox-PE
C.elegans-SE
Yeast-SE
E.coli-SE
Swinepox-SE
Datasets(36 bp)
30
25
20
15
10
5
0
Max RAM used(G)
SSAKE
VCAKE
QSRA
SHARCGS
Edena-strict
Edena-nonstrict
Velvet
SOAPdenovo
Taipan
NGS assembly programs

Figure 2. Computational running time and maximum memory occupancy of 36-mer short reads assembly procedures. (A) the computational times of each assembler for different datasets. (B) the maximum RAM used during the assembly process. No data is shown when the RAM is insufficient or the assembly tool is not suitable for the dataset.
doi:10.1371/journal.pone.0017915.g002

assembly tasks with fairly little computational time and memory usage.

Especially, SOAPdenovo runs in an extreme speed as the exploitation of threads parallelization, but may perform not well enough for small datasets due to the initial task allocation. At last, Taipan was proposed as the hybrid of string-based and graph-based approaches [17], with the dominative feature - the exceedingly short runtime. Nevertheless, the minimum RAM of computer to execute the assembler is high and the requirement for memory grows slowly with the increase of dataset size. Result also shows that more running time and RAM consuming are demanded for paired-end (PE) reads assembly than single-end (SE) reads dataset with the same assembler (Unpublished data). Compare with 36-mer short reads assembly, only OLC, *De Bruijn* and hybrid assemblers can be applied for 75-mer short reads assembly. Our study indicates that no significant difference on the computational time and RAM occupancy for the assembly of these two types of short fragments, with the same sequencing coverage.

The assembly accuracy and integrity is another consideration for the evaluation of the short reads assemblers. Obviously, contigs with high fidelity and genome coverage are our expectation. Different assemblers have their own performance. Their percentages of correctly mapped contigs and genome coverage for different datasets are shown in Figures 4 and 5. The latest version of SSAKE is of robustness to sequencing errors, compared with it is first version, which was introduced to handle error-free short reads [9]. Other string-based assemblers, such as VCAKE and SHARCGS performed in rivalry with the latest version of SSAKE while QSRA could only generate less precise and lower coverage contigs in contrast with previous tools. What deserves to be mentioned is that Edena, as an assembler based on the overlap-layout-consensus algorithm (OLC) [16], had a quite surpassing performance on various datasets. However, contigs produced from two *De Bruijn* graph-based assemblers, especially SOAPdenovo, were of lower accuracy, but with comparable genome coverage to string-based software. Nevertheless, when handling dataset of huge size, such as short reads from *C.elegans* genome, SOAPdenovo had similar performance as Edena. This result can be elucidated as following: for *De Bruijn* graph-based method, certain proportion of base errors are incorporated into contigs during the construction of graph with k mers generated from input short reads, this process then generate less precise contigs. In the end, the hybrid assembler, Taipan was capable to generate sequences of high accuracy and genome coverage as string-based assembler for small datasets, but performed poorly for the assembly of large genome dataset. After inspection on this assembly procedure, we supposed that it was the exploitation of only partial fraction of short reads that lead to the low coverage productive contigs. Here, we also verified that PE reads is superior to SE reads in terms of resolution for repetitive elements, which is in consistent with previous study [18]. In addition, our result shows that more accurate and higher genome coverage contigs can be produced with longer reads datasets, while it may

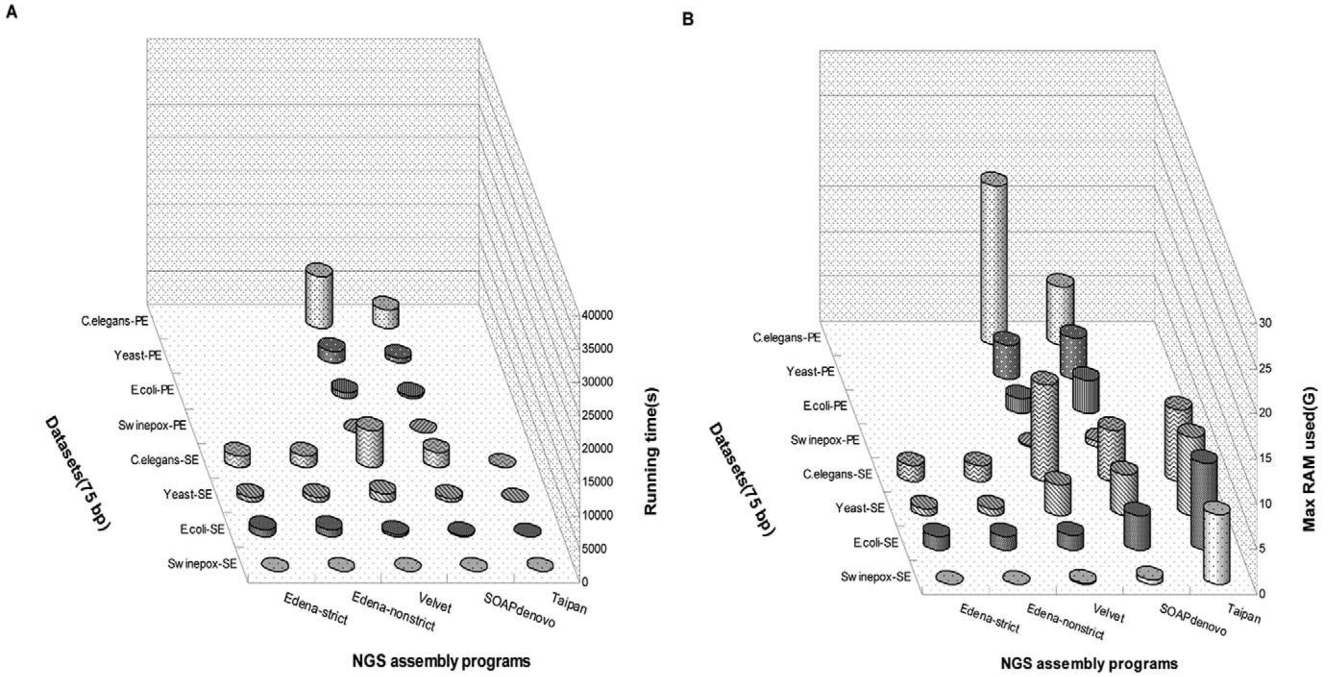


Figure 3. Computational running time and maximum memory occupancy of 75-mer short reads assembly procedures. (A) the computational times of each assembler for different datasets. (B) the maximum RAM used during the assembly process. No data is shown when the RAM is insufficient or the assembly tool is not suitable for the dataset.
doi:10.1371/journal.pone.0017915.g003

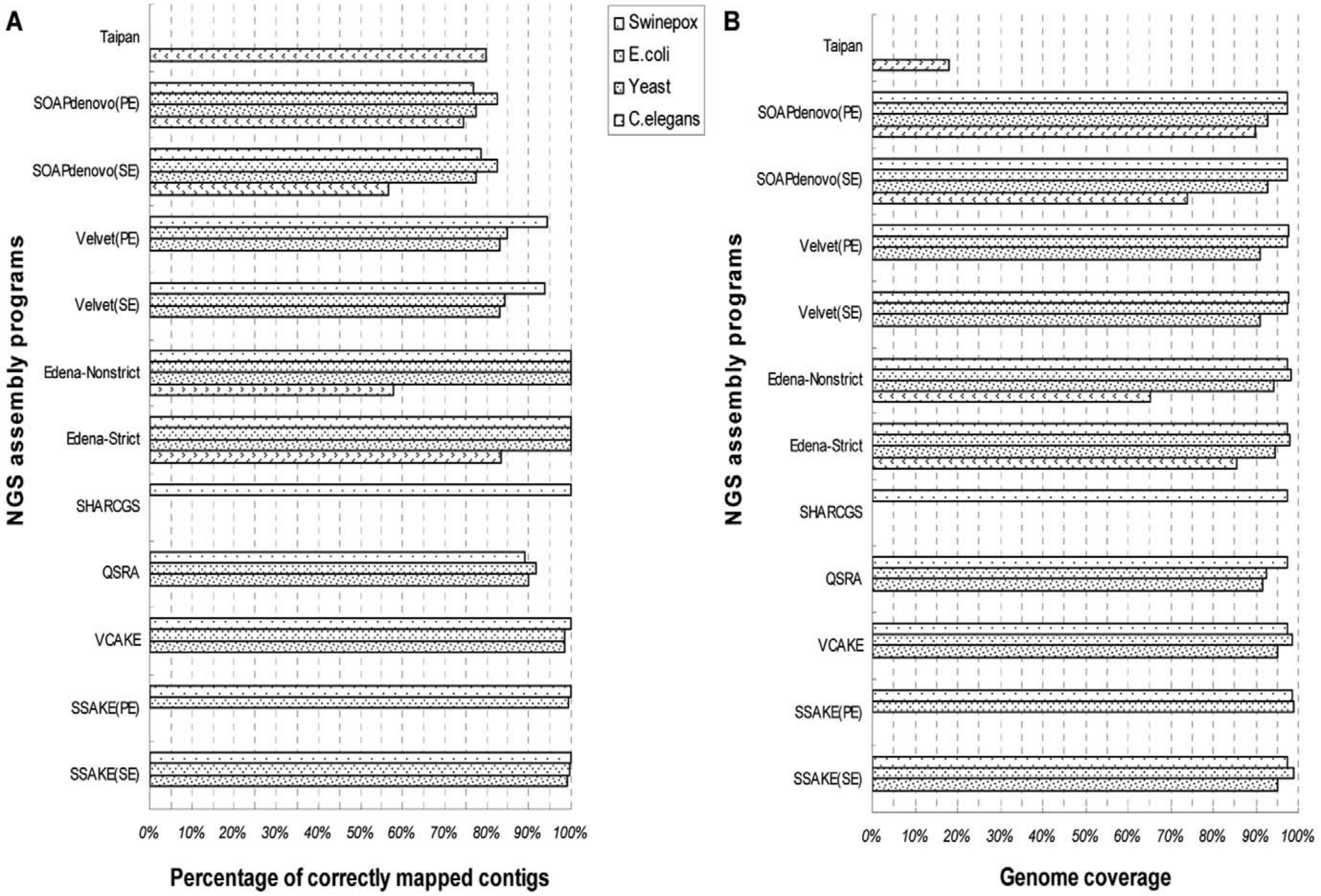


Figure 4. Accuracy and integrity for 36-mer datasets assembly. For short reads assembly, accurate and high genome coverage contigs are expected. Here, the quality of consequential contigs is shown with (A) the accuracy of assembled contigs and (B) the genome coverage of the assembled contigs. No data is shown when the RAM is insufficient or the assembly tool is not suitable for the dataset.
doi:10.1371/journal.pone.0017915.g004

be a paradox for assembly of large genome, such as *C.elegans*, of which none of the selected assemblers in this study is suitable for its 75-mer reads assembly.

For further analysis of assembled contigs, the contig size distribution was calculated and shown in Figures 6 and 7. For many biological studies, DNA sequence with sufficient length is necessary. Under ideal condition, only one contig that matches the whole genome sequence perfectly could be generated from each assembly procedure. Practically, the contigs generated by different assembly procedures are separated by gaps for the presence of repetitive fragments. From Figures 2, 3, 4, 5, 6, and 7, it is clear that different assembly strategies perform differently on diverse datasets. For dataset of very small size, string-based assemblers produced fewer but longer reads than *De Bruijn* graph-based tools. However, it became reverse when the size of dataset increases. Edena, the OLC assembler, could assemble short reads into relatively long contigs for various datasets. Taipan, as a hybrid assembly tool, had better performance than Edena for small datasets. When handling short fragments from large genomes such as *C.elegans*, even though fairly longer largest contigs was formed, N50 and N80 size were not available with too few assembled contigs. Here in general, we can claim that PE reads or longer reads would generate better assembly results. Besides, for *De Bruijn* assemblers, Velvet produced better assembly result than SOAPdenovo when assembly of 75-mer short reads datasets, because of the wider range of K value to be chosen in Velvet.

Discussion

Even though the assembly algorithms for *de novo* genome assembly have been well-reviewed [19], we are the first to test and compare these tools with different datasets practically. The key concern for the assessment of an assembler is its usability and assembly quality. We evaluated the current assemblers from the two aspects with simulated Solexa short reads datasets (the detail could be found in the method section) on one single server machine, as Solexa/IIIumina sequencing technology is the most widely applied technology. Our results show that string-based assemblers, OLC assemblers are well-suited for very short reads and longer reads respectively for small genome comprising millions of short reads, when the computational power is limited. But Taipan is a better choice for its excellent assembly speed if the RAM of machine is sufficient. For large datasets of more than hundred millions of short reads, *De Bruijn* graph-based assemblers could have commendable resolutions due to their short runtime and low RAM occupancy, of which SOAPdenovo performs well on very short reads, while ALLPATHS-LG could be a good choice for ~100 bp short reads assembly, as it was described [6]. In terms of ease of software installation, string-based assemblers and hybrid

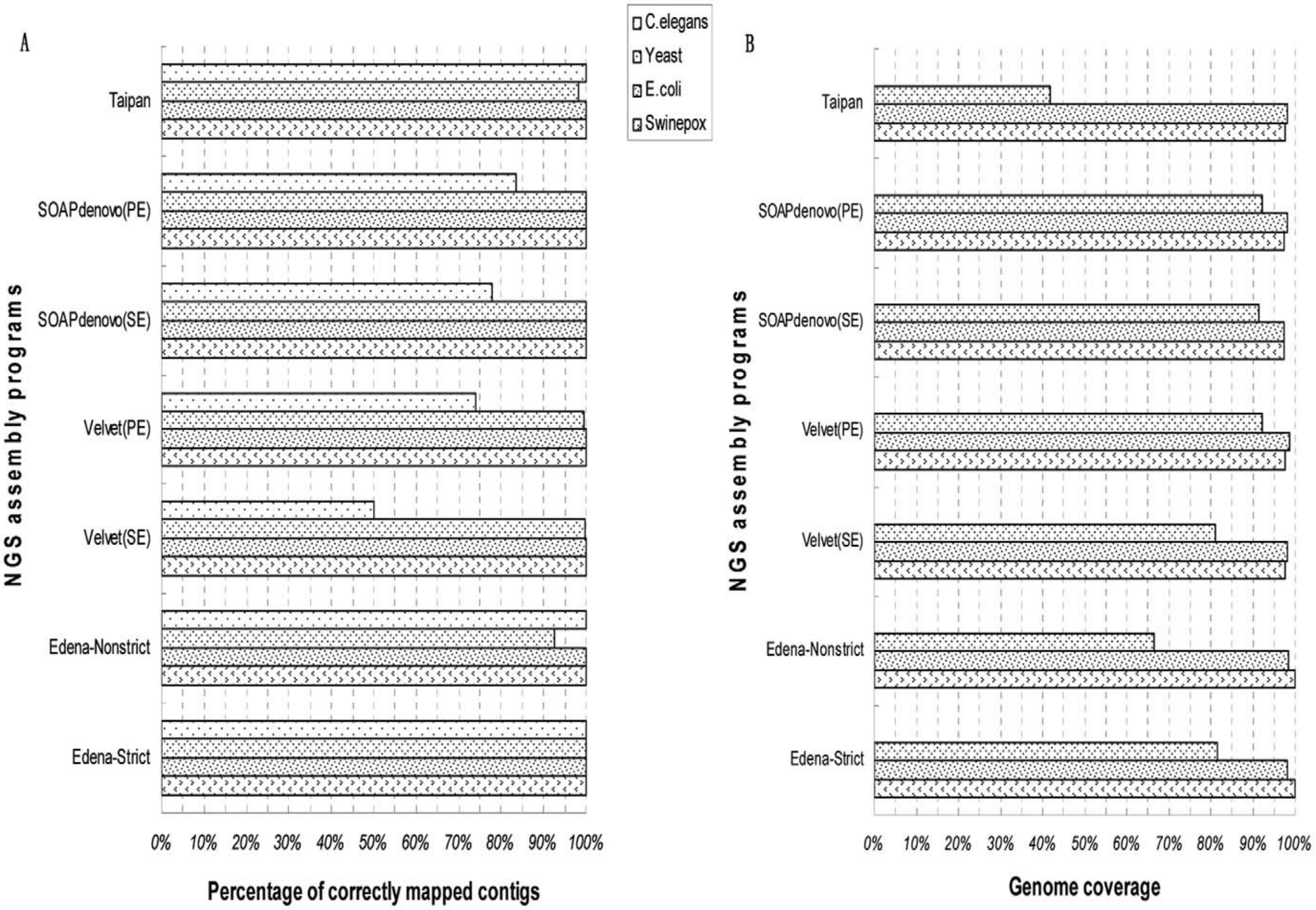


Figure 5. Accuracy and integrity for 75-mer datasets assembly. For short reads assembly, accurate and high genome coverage contigs are expected. Here, the quality of consequential contigs is shown with (A) the accuracy of assembled contigs and (B) the genome coverage of the assembled contigs. No data is shown when the RAM is insufficient or the assembly tool is not suitable for the dataset.
doi:10.1371/journal.pone.0017915.g005

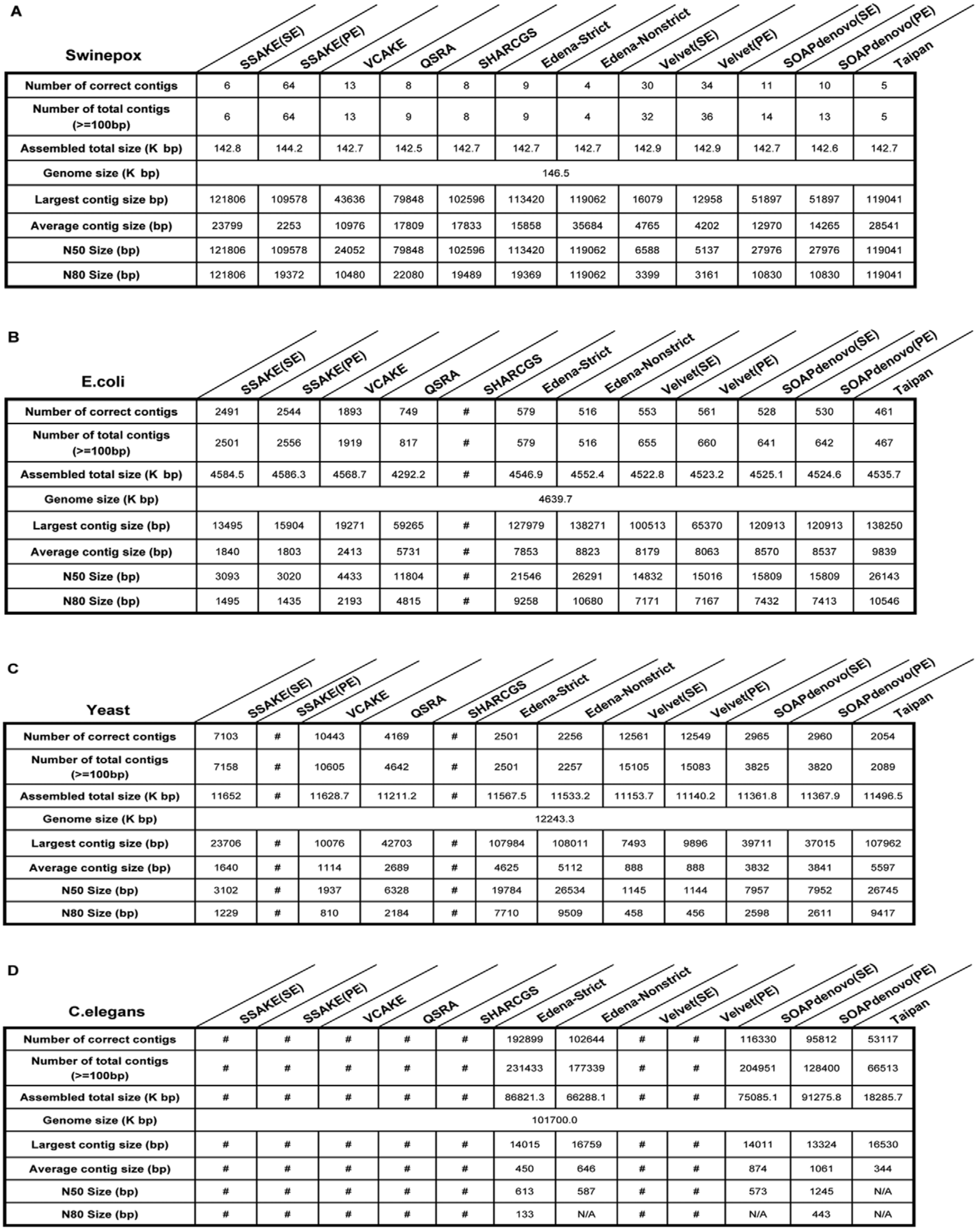
assembler are superior to graph-based ones, of which SOAPdenovo is complex for the creation of configuration file. In addition, as shown in this study, new assemblers for longer reads are much-needed, since the majority ones were designed for very short reads. Recommended assemblers for different assembly processes are shown in Table 1.

Assembly for small genomes, such as prokaryote organisms, has been well resolved [20,21,22]. However, short reads from eukaryote genomes, with features of gargantuan size and high repetitiveness, make sky-high requirement for assembly strategies and computer hardware [10,11,23,24]. Exceptional data storage methods are required to reduce RAM occupancy, for example, ABySS transfers the sequence reads into binary format to save the computational space [12]. Threads parallelization is a solution to accelerate assembly speed. Three hierarchies of parallelization are taken into implementation: multi-thread on a single machine [11], multi-process with cluster machines [12,25] and cloud computation (http://sourceforge.net/apps/mediawiki/contrail-bio/index.php?title = Contrail). Interestingly, GPU computational method has been applied in other two short reads analysis procedures, *i.e.* error correction and alignment, and speeded up of these processes many times as reported [26,27,28]. Great improvement may be expected with the application of this approach in assembly process afterwards. Besides, integration of multi datasets from various sequencing strategies are exploited to tackle the complex genome assembly [29], which greatly challenge the development of assembly algorithms to suit for diverse short reads. Usually, several assembler are combined for this issue [30]. Meanwhile, the accuracy and read length of sequenced tags are increasing stepwisely, and PE sequencing strategies are extensively carried out on different next generation sequencing (NGS) platforms. With the cooperation between biologist, bioinformaticians and developers of high performance machine, we can expect that *de novo* assembly of short reads will be less challenging for NGS data analysis in the near future.

Methods

Short reads data simulation

To get the precise information about the quality of assembled results, we simulated the short reads datasets sequencing from Solexa/IIIumina with the perl script program (see Package S1), for the reason that there is no exact genome sequence for real sequenced datasets. Currently, the real data from Solexa platform is 75 bp per read, while the 36 bp sequencing mode is still well-supported. According to the report by Jay Shendure & Hanlee [1], the dominant error type for Solexa sequencing protocol is substitution, and the error rate of 0.1% could



A

Swinepox	SSAKE(SE)	SSAKE(PE)	VCAKE	QSRA	SHARCGS	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	6	64	13	8	8	9	4	30	34	11	10	5
Number of total contigs (>=100bp)	6	64	13	9	8	9	4	32	36	14	13	5
Assembled total size (K bp)	142.8	144.2	142.7	142.5	142.7	142.7	142.7	142.9	142.9	142.7	142.6	142.7
Genome size (K bp)	146.5											
Largest contig size bp)	121806	109578	43636	79848	102596	113420	119062	16079	12958	51897	51897	119041
Average contig size (bp)	23799	2253	10976	17809	17833	15858	35684	4765	4202	12970	14265	28541
N50 Size (bp)	121806	109578	24052	79848	102596	113420	119062	6588	5137	27976	27976	119041
N80 Size (bp)	121806	19372	10480	22080	19489	19369	119062	3399	3161	10830	10830	119041

B

E.coli	SSAKE(SE)	SSAKE(PE)	VCAKE	QSRA	SHARCGS	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	2491	2544	1893	749	#	579	516	553	561	528	530	461
Number of total contigs (>=100bp)	2501	2556	1919	817	#	579	516	655	660	641	642	467
Assembled total size (K bp)	4584.5	4586.3	4568.7	4292.2	#	4546.9	4552.4	4522.8	4523.2	4525.1	4524.6	4535.7
Genome size (K bp)	4639.7											
Largest contig size (bp)	13495	15904	19271	59265	#	127979	138271	100513	65370	120913	120913	138250
Average contig size (bp)	1840	1803	2413	5731	#	7853	8823	8179	8063	8570	8537	9839
N50 Size (bp)	3093	3020	4433	11804	#	21546	26291	14832	15016	15809	15809	26143
N80 Size (bp)	1495	1435	2193	4815	#	9258	10680	7171	7167	7432	7413	10546

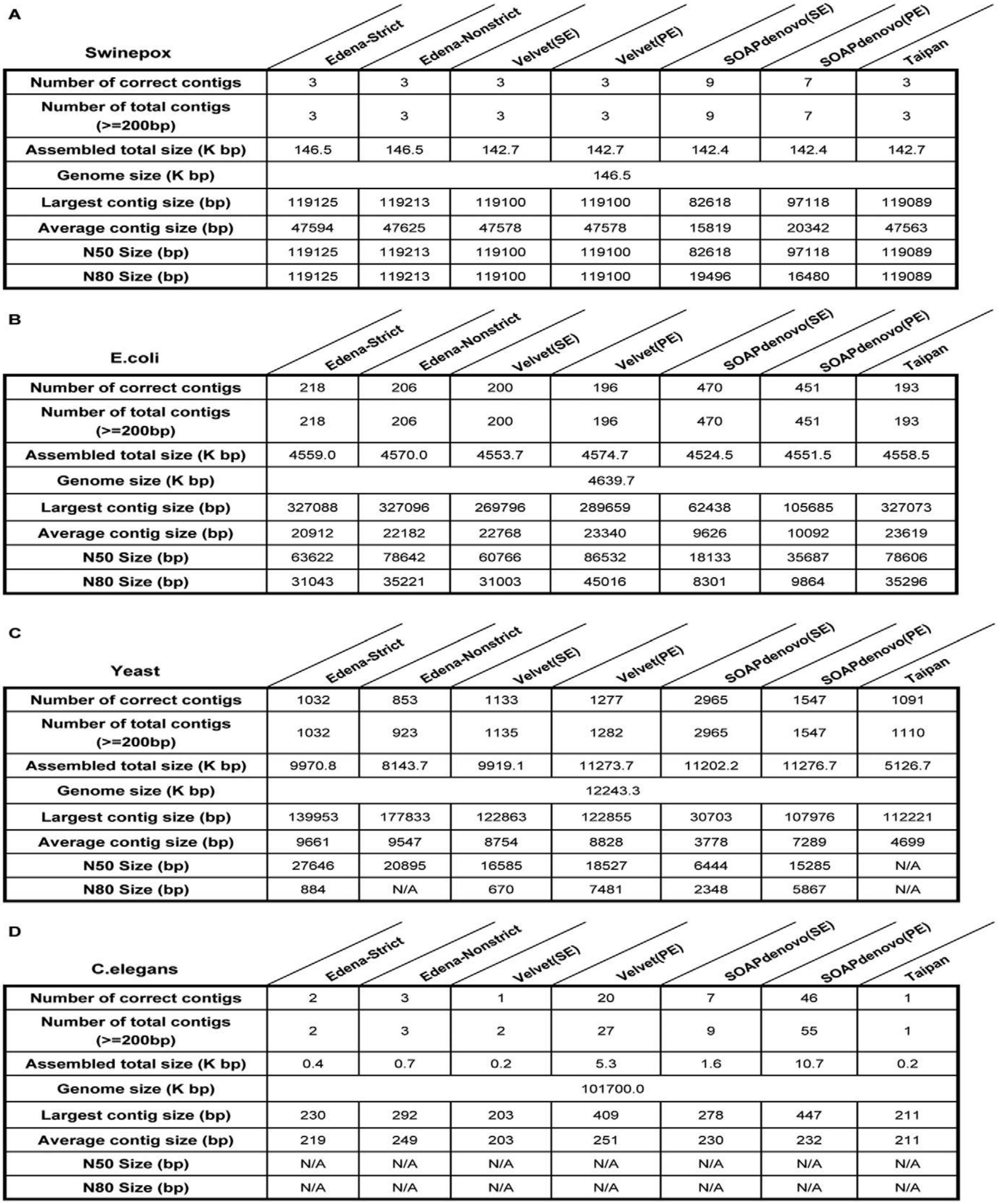
C

Yeast	SSAKE(SE)	SSAKE(PE)	VCAKE	QSRA	SHARCGS	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	7103	#	10443	4169	#	2501	2256	12561	12549	2965	2960	2054
Number of total contigs (>=100bp)	7158	#	10605	4642	#	2501	2257	15105	15083	3825	3820	2089
Assembled total size (K bp)	11652	#	11628.7	11211.2	#	11567.5	11533.2	11153.7	11140.2	11361.8	11367.9	11496.5
Genome size (K bp)	12243.3											
Largest contig size (bp)	23706	#	10076	42703	#	107984	108011	7493	9896	39711	37015	107962
Average contig size (bp)	1640	#	1114	2689	#	4625	5112	888	888	3832	3841	5597
N50 Size (bp)	3102	#	1937	6328	#	19784	26534	1145	1144	7957	7952	26745
N80 Size (bp)	1229	#	810	2184	#	7710	9509	458	456	2598	2611	9417

D

C.elegans	SSAKE(SE)	SSAKE(PE)	VCAKE	QSRA	SHARCGS	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	#	#	#	#	#	192899	102644	#	#	116330	95812	53117
Number of total contigs (>=100bp)	#	#	#	#	#	231433	177339	#	#	204951	128400	66513
Assembled total size (K bp)	#	#	#	#	#	86821.3	66288.1	#	#	75085.1	91275.8	18285.7
Genome size (K bp)	101700.0											
Largest contig size (bp)	#	#	#	#	#	14015	16759	#	#	14011	13324	16530
Average contig size (bp)	#	#	#	#	#	450	646	#	#	874	1061	344
N50 Size (bp)	#	#	#	#	#	613	587	#	#	573	1245	N/A
N80 Size (bp)	#	#	#	#	#	133	N/A	#	#	N/A	443	N/A

Figure 6. Statistics for assembled contigs of 36-mer short reads. Indicatrix that illustrates the feature of size distribution are adopted for analysis. "#" denotes the RAM of machine is not enough, and "N/A" means the data is not available. The N50 size and N80 size represent the maximum read length for which all contigs greater than or equal to the threshold covered 50% or 80% of the reference genome.
doi:10.1371/journal.pone.0017915.g006



A

Swinepox	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	3	3	3	3	9	7	3
Number of total contigs (>=200bp)	3	3	3	3	9	7	3
Assembled total size (K bp)	146.5	146.5	142.7	142.7	142.4	142.4	142.7
Genome size (K bp)	146.5						
Largest contig size (bp)	119125	119213	119100	119100	82618	97118	119089
Average contig size (bp)	47594	47625	47578	47578	15819	20342	47563
N50 Size (bp)	119125	119213	119100	119100	82618	97118	119089
N80 Size (bp)	119125	119213	119100	119100	19496	16480	119089

B

E.coli	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	218	206	200	196	470	451	193
Number of total contigs (>=200bp)	218	206	200	196	470	451	193
Assembled total size (K bp)	4559.0	4570.0	4553.7	4574.7	4524.5	4551.5	4558.5
Genome size (K bp)	4639.7						
Largest contig size (bp)	327088	327096	269796	289659	62438	105685	327073
Average contig size (bp)	20912	22182	22768	23340	9626	10092	23619
N50 Size (bp)	63622	78642	60766	86532	18133	35687	78606
N80 Size (bp)	31043	35221	31003	45016	8301	9864	35296

C

Yeast	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	1032	853	1133	1277	2965	1547	1091
Number of total contigs (>=200bp)	1032	923	1135	1282	2965	1547	1110
Assembled total size (K bp)	9970.8	8143.7	9919.1	11273.7	11202.2	11276.7	5126.7
Genome size (K bp)	12243.3						
Largest contig size (bp)	139953	177833	122863	122855	30703	107976	112221
Average contig size (bp)	9661	9547	8754	8828	3778	7289	4699
N50 Size (bp)	27646	20895	16585	18527	6444	15285	N/A
N80 Size (bp)	884	N/A	670	7481	2348	5867	N/A

D

C.elegans	Edena-Strict	Edena-Nonstrict	Velvet(SE)	Velvet(PE)	SOAPdenovo(SE)	SOAPdenovo(PE)	Taipan
Number of correct contigs	2	3	1	20	7	46	1
Number of total contigs (>=200bp)	2	3	2	27	9	55	1
Assembled total size (K bp)	0.4	0.7	0.2	5.3	1.6	10.7	0.2
Genome size (K bp)	101700.0						
Largest contig size (bp)	230	292	203	409	278	447	211
Average contig size (bp)	219	249	203	251	230	232	211
N50 Size (bp)	N/A	N/A	N/A	N/A	N/A	N/A	N/A
N80 Size (bp)	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Figure 7. Statistics for assembled contigs of 75-mer short reads. Indicatrix that illustrates the feature of size distribution are adopted for analysis. "#" denotes the RAM of machine is not enough, and "N/A" means the data is not available. The N50 size and N80 size represent the maximum read length for which all contigs greater than or equal to the threshold covered 50% or 80% of the reference genome.

doi:10.1371/journal.pone.0017915.g007

Table 1. Recommended assemblers for different genome assembly[1].

	Type of reads	RAM of Machine	Recommended assembler
Small genome (Microorganism)	Very short (36 bp)	Large (>16G)	Hybrid assembler: Taipan
		Small (<16G)	SSAKE, QSRA, Edena
	Short (75 bp)	Large (>16G)	Hybrid assembler: Taipan
		Small (<16G)	OLC assembler: Edena
Large genome (Eukaryote)	Very short (36 bp)	Large (>16G)	De Bruijn assembler: SOAPdenovo
		Small (<16G)	—
	Short (75 bp)	Large (>16G)	De Bruijn assembler: ALLPATHS-LG
		Small (<16G)	—

[1] According to our evaluation study, the specific assembler is recommended for different type of assembly procedure. Herein, only tools running on a single machine are considered, while other assemblers running on cluster machines, such as ABySS and Ray, may also perform well for large genome assembly.

doi:10.1371/journal.pone.0017915.t001

be achieved after quality filtering. Hence, in this work, we only consider substitution error type and adopt 0.1% error rate, even though which may change slightly as sequencing technology develops. The Swinepox virus (Swinepox), *Escherichia coli* str. K-12 substr (*E.coli*), *Saccharomyces cerevisiae* (Yeast) and *Caenorhabditis elegans* (*C.elegans*) genomes were downloaded from Genebank (Genebank accession number NC_003389, NC_000913, NC_001133–NC_001148, NC_003279–NC_003284) respectively. SE reads dataset and PE reads dataset with length of 36 bp and 75 bp were simulated according to each genome sequence. For SE reads, all possible 36mers (or 75 mers) were extracted from both strands for these genomes then added an error rate of 0.1% to the generated reads. Sequences were selected at random to simulate up to 100× read coverage for the first three genomes and up to 50× coverage for *C.elegans* genome. For PE reads, simulated sequences were generated by sliding window approach with an (R+2r) bp window size and 1 bp step size (R is 2000 for *C.elegans*, 500 for 3 other genomes, r is the short read size). Along each genome reference sequence the first 36 bp (or 75 bp) and the reverse complement of the last 36 bp (or 75 bp) in each window frame were collected than add an error rate of 0.1% to the reads. PE read datasets with the same read coverage as from SE reads synthesis procedure were generated. The size comparison of these datasets is shown in Figure 8; Figure 9 displays the pipeline of the whole evaluation study.

Preliminary analysis of reference genome sequence

Genome sequence assembly is greatly challenged by repeat sequences, especially when the repeats are longer than short reads [4]. To address this issue, longer reads and PE information are required [13]. Before the assembly procedure implemented, we detected the repeat elements in reference genomes with Tandem Repeats Finder [31]. The number of repeats reflects the complexity of target sequence to a large extent (see Figure 10).

Program implementation

Eight short reads assembly programs (see Figure 11), which represent 4 different assembly strategies, were selected for assembly of simulated short reads. For each assembly procedure, we set 3 different series of parameters (Table S2), from which the best assembly result was chosen for the evaluation of the performance of each program respectively.

All the selected programs were run on a server machine equipped with four 2.4GHz Intel(R) Xeon(R) 4 CPU, 4 cores within each CPU, and 32 GB of RAM. The operating system is Ubuntu 8.04.4 with version of X_86 64 bit.

Performance evaluation

The computational time consuming and maximum memory occupancy during each assembly procedure are recorded with a perl and python script. The mean of the computational time and

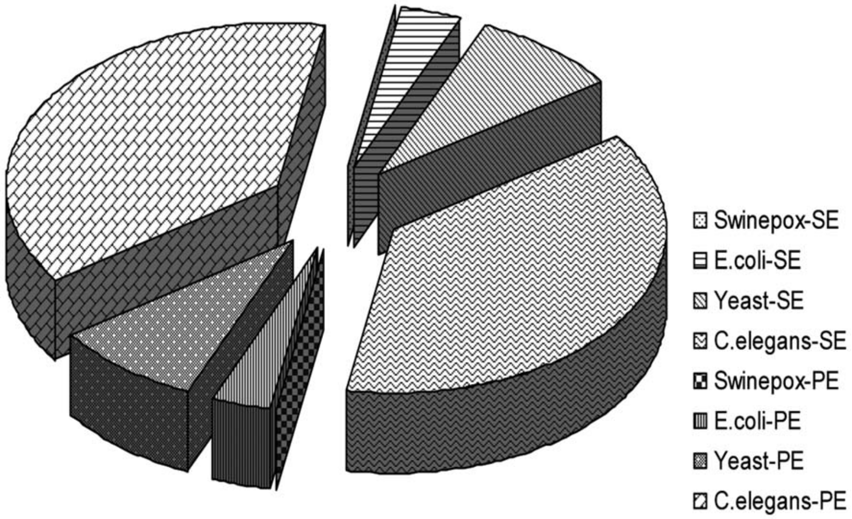


Figure 8. Size comparison of datasets used in this study. This figure shows the relative size comparison of short reads datasets with different legends. SE denotes Single-end short reads dataset, while PE denotes Paired-end short reads dataset.

doi:10.1371/journal.pone.0017915.g008

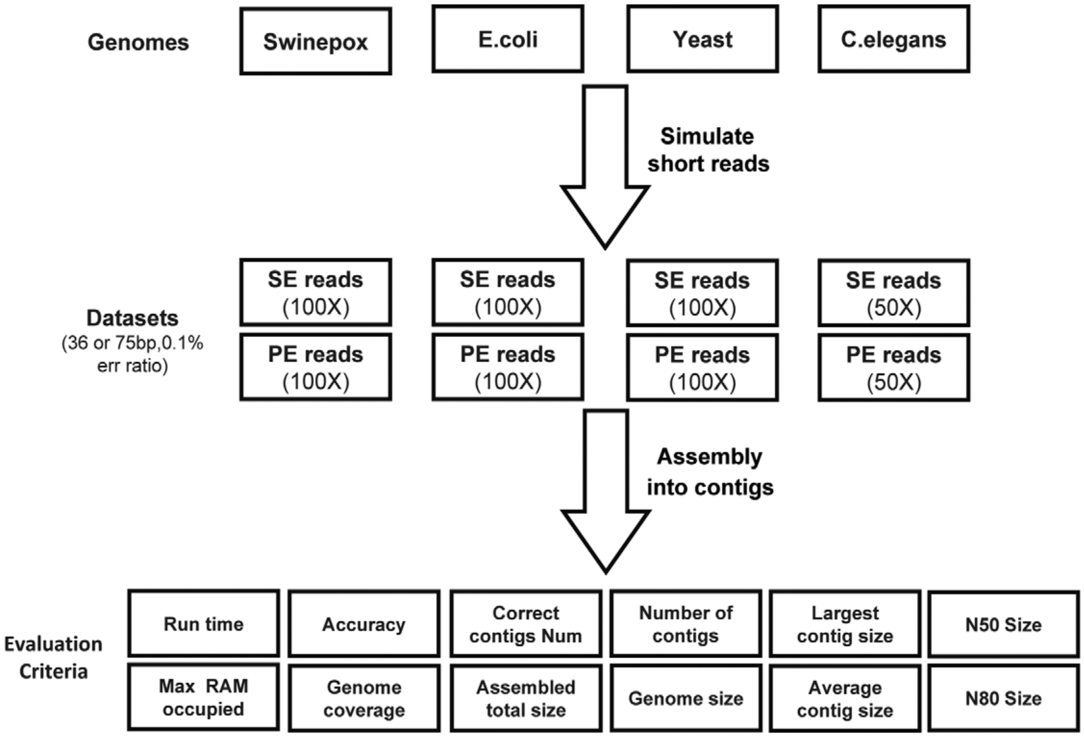
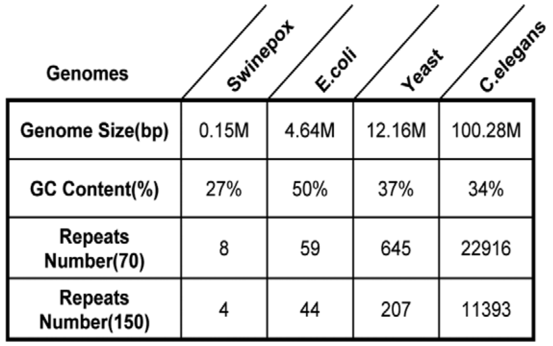


Figure 9. Pipeline for evaluation of short reads assembly programs. Four reference genomes with different size are exploited to generate short reads bearing base errors. The performance of assemblers is evaluated through computational time, accuracy, integrity and contig size, etc.
doi:10.1371/journal.pone.0017915.g009

computational memory cost of three processes with different parameters was considered to be the performance of each corresponding assembler. Data was not shown when the machine memory is insufficient or the assembler is not suitable for the dataset.

Accuracy and integrity

Contigs generated from each assembly process were mapped to each homologous genome reference sequence with NCBI Blast-2.2.20 for Windows 32bit machine [32], of which with size no shorter than Ybp (Y is 100 bp for 36mer datasets and 200 bp for 75mer datasets) and at least 98% of each read completely match to the reference sequence were presumed to be correct. We calculate the accuracy with $Acc = \mathrm{NC/N}$, where NC and N represent the number of correct contigs and the number of contigs longer than Ybp respectively. The integrity was computed with equation $Inte = (\sum_{i=1}^{Nc} Li)/\mathrm{L}$, which means the ratio of the sum of all the correct contig sizes to the reference sequence size. Outcome with optimal accuracy and integrity was chosen as the best performance of each assembler.



Genomes	Swinepox	E.coli	Yeast	C.elegans
Genome Size(bp)	0.15M	4.64M	12.16M	100.28M
GC Content(%)	27%	50%	37%	34%
Repeats Number(70)	8	59	645	22916
Repeats Number(150)	4	44	207	11393

Figure 10. Analysis of complexity of reference genome sequences. Tandem repeats finder (Version 4.04) is utilized to detect the number of repeat elements with length less than 2000 bp, the parameter "minimum alignment score " is set to 70 and 150 for two types of short reads. The increase of genome size, repeat numbers and GC content may imply the increasing in genome assembly complexity.
doi:10.1371/journal.pone.0017915.g010

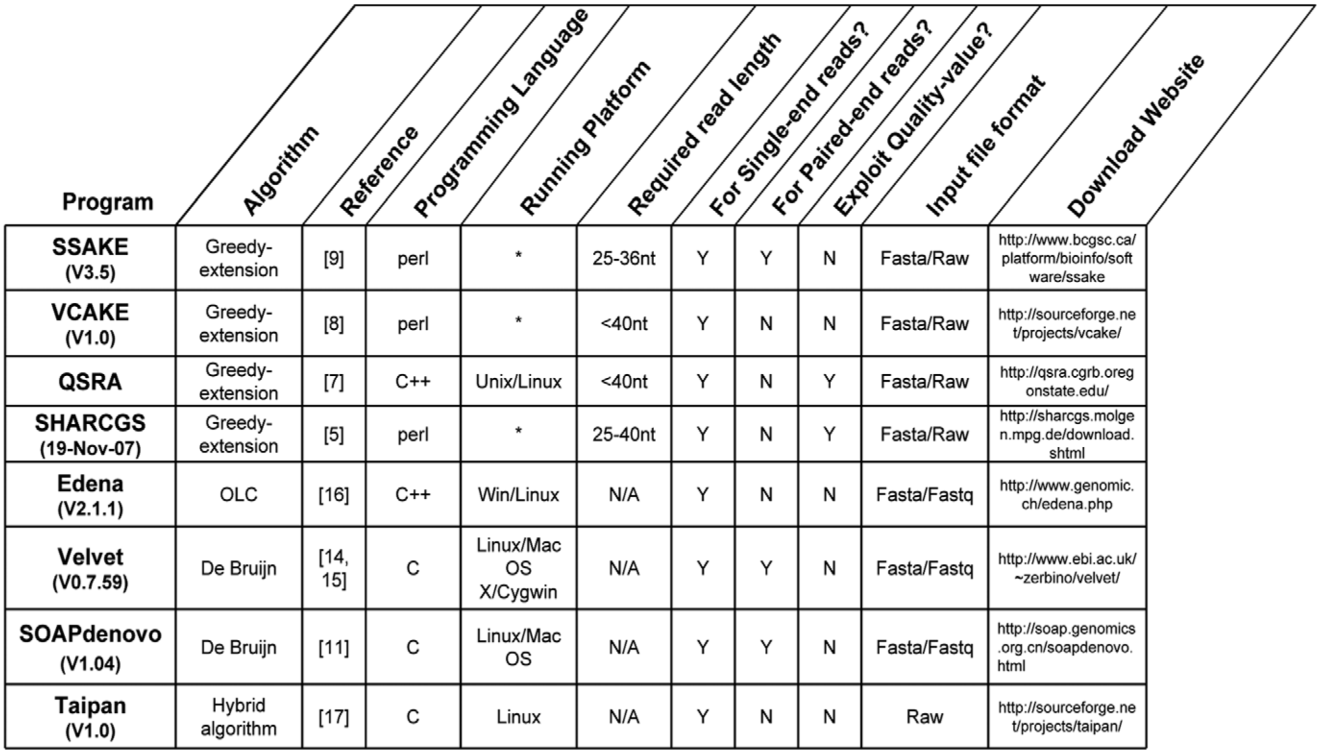
Statistical information of assembled contigs

To further evaluate the performance of each assembly tool, we also provide the information of size distribution of assembled contigs, including number of correct contigs, number of total assembled contigs, largest contig size, average contig size, N50 and N80 contig sizes. The N50 and N80 represent the size N such that 50% or 80% of the genome is contained in contigs of size N or greater. With this information, we can compare and measure the genome assemblies statistically.

Supporting Information

Table S1 The websites and references for de *novo* NGS assemblers.
(DOC)

Table S2 The parameters for each assembly procedure. For each pair of assembler and dataset, 3 groups of parameter are adopted for short reads assembly. Symbol "—" and "*Out of RAM*" means the assembler does not suit for corresponding type of



Program	Algorithm	Reference	Programming Language	Running Platform	Required read length	For Single-end reads?	For Paired-end reads?	Exploit Quality-value?	Input file format	Download Website
SSAKE (V3.5)	Greedy-extension	[9]	perl	*	25-36nt	Y	Y	N	Fasta/Raw	http://www.bcgsc.ca/platform/bioinfo/software/ssake
VCAKE (V1.0)	Greedy-extension	[8]	perl	*	<40nt	Y	N	N	Fasta/Raw	http://sourceforge.net/projects/vcake/
QSRA	Greedy-extension	[7]	C++	Unix/Linux	<40nt	Y	N	Y	Fasta/Raw	http://qsra.cgrb.oregonstate.edu/
SHARCGS (19-Nov-07)	Greedy-extension	[5]	perl	*	25-40nt	Y	N	Y	Fasta/Raw	http://sharcgs.molgen.mpg.de/download.shtml
Edena (V2.1.1)	OLC	[16]	C++	Win/Linux	N/A	Y	N	N	Fasta/Fastq	http://www.genomic.ch/edena.php
Velvet (V0.7.59)	De Bruijn	[14, 15]	C	Linux/Mac OS X/Cygwin	N/A	Y	Y	N	Fasta/Fastq	http://www.ebi.ac.uk/~zerbino/velvet/
SOAPdenovo (V1.04)	De Bruijn	[11]	C	Linux/Mac OS	N/A	Y	Y	N	Fasta/Fastq	http://soap.genomics.org.cn/soapdenovo.html
Taipan (V1.0)	Hybrid algorithm	[17]	C	Linux	N/A	Y	N	N	Raw	http://sourceforge.net/projects/taipan/

Figure 11. Features of selected short reads assembly programs. Noncommercial programs based on varied sorts of assembly approaches were selected for testing on synthetic Solexa short reads. "*" indicates any operating systems with perl interpreter, while "OLC" is for overlap-layout-consensus, and "N/A" for not available. The features for different programs are obtained from related references and documents of the latest version software (version information is not listed here).
doi:10.1371/journal.pone.0017915.g011

dataset and memory required for the assembly process is beyond computer power, while parameters in bold will be the best for the assembly.
(DOC)

Package S1 The perl scripts and the test file used to simulate the short reads . We modified the program written by Juliane Dohm and Claudio Lottaz to simulate both single-end reads and paired-end reads from a given reference sequence.
(RAR)

Acknowledgments

We thank Juliane Dohm and Claudio Lottaz for the idea and scripts about simulation of short reads datasets.

Author Contributions

Conceived and designed the experiments: BS. Performed the experiments: WZ JC YY YT JS. Analyzed the data: WZ JC YY YT JS BS. Contributed reagents/materials/analysis tools: WZ JC YY YT JS. Wrote the paper: WZ JC BS.

References

1. Shendure J, Ji H (2008) Next-generation DNA sequencing. Nat Biotechnol 26: 1135–1145.
2. Pevzner PA, Tang H, Waterman MS (2001) An Eulerian path approach to DNA fragment assembly. Proc Natl Acad Sci U S A 98: 9748–9753.
3. Pop M, Kosack D (2004) Using the TIGR assembler in shotgun sequencing projects. Methods Mol Biol 255: 279–294.
4. Miller JR, Koren S, Sutton G (2010) Assembly algorithms for next-generation sequencing data. Genomics 95: 315–327.
5. Dohm JC, Lottaz C, Borodina T, Himmelbauer H (2007) SHARCGS, a fast and highly accurate short-read assembly algorithm for *de novo* genomic sequencing. Genome Res 17: 1697–1706.
6. Gnerre S, Maccallum I, Przybylski D, Ribeiro FJ, Burton JN, et al. (2010) High-quality draft assemblies of mammalian genomes from massively parallel sequence data. Proc Natl Acad Sci U S A.
7. Bryant DW, Jr., Wong WK, Mockler TC (2009) QSRA: a quality-value guided *de novo* short read assembler. BMC Bioinformatics 10: 69.
8. Jeck WR, Reinhardt JA, Baltrus DA, Hickenbotham MT, Magrini V, et al. (2007) Extending assembly of short DNA sequences to handle error. Bioinformatics 23: 2942–2944.
9. Warren RL, Sutton GG, Jones SJ, Holt RA (2007) Assembling millions of short DNA sequences using SSAKE. Bioinformatics 23: 500–501.
10. Li R, Fan W, Tian G, Zhu H, He L, et al. (2010) The sequence and *de novo* assembly of the giant panda genome. Nature 463: 311–317.
11. Li R, Zhu H, Ruan J, Qian W, Fang X, et al. (2010) *De novo* assembly of human genomes with massively parallel short read sequencing. Genome Res 20: 265–272.
12. Simpson JT, Wong K, Jackman SD, Schein JE, Jones SJ, et al. (2009) ABySS: a parallel assembler for short read sequence data. Genome Res 19: 1117–1123.
13. Cahill MJ, Koser CU, Ross NE, Archer JA (2010) Read length and repeat resolution: exploring prokaryote genomes using next-generation sequencing technologies. PLoS One 5: e11518.
14. Zerbino DR, Birney E (2008) Velvet: algorithms for *de novo* short read assembly using de Bruijn graphs. Genome Res 18: 821–829.
15. Zerbino DR, McEwen GK, Margulies EH, Birney E (2009) Pebble and rock band: heuristic resolution of repeats and scaffolding in the velvet short-read *de novo* assembler. PLoS One 4: e8407.
16. Hernandez D, Francois P, Farinelli L, Osteras M, Schrenzel J (2008) *De novo* bacterial genome sequencing: millions of very short reads assembled on a desktop computer. Genome Res 18: 802–809.
17. Schmidt B, Sinha R, Beresford-Smith B, Puglisi SJ (2009) A fast hybrid short read fragment assembly algorithm. Bioinformatics 25: 2279–2280.
18. Chaisson MJ, Brinza D, Pevzner PA (2009) *De novo* fragment assembly with short mate-paired reads: Does the read length matter? Genome Res 19: 336–346.
19. Paszkiewicz K, Studholme DJ (2010) De novo assembly of short sequence reads. Brief Bioinform 11: 457–472.

20. Farrer RA, Kemen E, Jones JD, Studholme DJ (2009) *De novo* assembly of the Pseudomonas syringae pv. syringae B728a genome using Illumina/Solexa short sequence reads. FEMS Microbiol Lett 291: 103–111.
21. Kingsford C, Schatz MC, Pop M (2010) Assembly complexity of prokaryotic genomes using short reads. BMC Bioinformatics 11: 21.
22. Nishito Y, Osana Y, Hachiya T, Popendorf K, Toyoda A, et al. (2010) Whole genome assembly of a natto production strain Bacillus subtilis natto from very short read data. BMC Genomics 11: 243.
23. Imelfort M, Edwards D (2009) *De novo* sequencing of plant genomes using second-generation technologies. Brief Bioinform 10: 609–618.
24. Nowrousian M, Stajich JE, Chu M, Engh I, Espagne E, et al. (2010) *De novo* assembly of a 40 Mb eukaryotic genome from short sequence reads: Sordaria macrospora, a model organism for fungal morphogenesis. PLoS Genet 6: e1000891.
25. Boisvert S, Laviolette F, Corbeil J (2010) Ray: simultaneous assembly of reads from a mix of high-throughput sequencing technologies. J Comput Biol 17: 1519–1533.
26. Schatz MC, Trapnell C, Delcher AL, Varshney A (2007) High-throughput sequence alignment using Graphics Processing Units. BMC Bioinformatics 8: 474.
27. Shi H, Schmidt B, Liu W, Muller-Wittig W (2010) A parallel algorithm for error correction in high-throughput short-read data on CUDA-enabled graphics hardware. J Comput Biol 17: 603–615.
28. Trapnell C, Schatz MC (2009) Optimizing Data Intensive GPGPU Computations for DNA Sequence Alignment. Parallel Comput 35: 429–440.
29. Diguistini S, Liao NY, Platt D, Robertson G, Seidel M, et al. (2009) *De novo* genome sequence assembly of a filamentous fungus using Sanger, 454 and Illumina sequence data. Genome Biol 10: R94.
30. Reinhardt JA, Baltrus DA, Nishimura MT, Jeck WR, Jones CD, et al. (2009) *De novo* assembly using low-coverage short read sequence data from the rice pathogen Pseudomonas syringae pv. oryzae. Genome Res 19: 294–305.
31. Benson G (1999) Tandem repeats finder: a program to analyze DNA sequences. Nucleic Acids Res 27: 573–580.
32. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215: 403–410.