Published online 29 April 2008

Nucleic Acids Research, 2008, Vol. 36, No. 10 ***3455–3462***
doi:10.1093/nar/gkn168

A new pheromone trail-based genetic algorithm for comparative genome assembly

Fangqing Zhao[1], Fanggeng Zhao[2], Tao Li[1] and Donald A. Bryant[1,*]

[1]Department of Biochemistry and Molecular Biology, The Pennsylvania State University, University Park, PA, 16802, USA and [2]Institute of Vehicle Management, Bengbu, 233011, China

Received November 29, 2007; Revised February 28, 2008; Accepted March 24, 2008

ABSTRACT

Gap closing is considered one of the most challenging and time-consuming tasks in bacterial genome sequencing projects, especially with the emergence of new sequencing technologies, such as pyrosequencing, which may result in large amounts of data without the benefit of large insert libraries for contig scaffolding. We propose a novel algorithm to align contigs with more than one reference genome at a time. This approach can successfully overcome the limitations of low degrees of conserved gene order for the reference and target genomes. A pheromone trail-based genetic algorithm (PGA) was used to search globally for the optimal placement for each contig. Extensive testing on simulated and real data sets shows that PGA significantly outperforms previous methods, especially when assembling genomes that are only moderately related. An extended version of PGA can predict additional candidate connections for each contig and can thus increase the likelihood of identifying the correct arrangement of each contig. The software and test data sets can be accessed at http://sourceforge.net/projects/pga4genomics/.

INTRODUCTION

Despite the fact that the assembly of bacterial genomes has become a routine task and multiple assemblers have been developed, the assembly problem is far from being solved. Gap closure is still considered the most challenging, time-consuming and laborious phase in finishing the sequence of a genome, especially with the emergence of new sequencing technologies. Pyrosequencing, which does not require any DNA cloning or library construction, can generate enormous amounts of data and can greatly reduce the cost of sequencing whole genomes. Such characteristics, however, may also increase the difficulty of subsequent contig assembly. After initial assembly from reads to contigs has occurred, traditional sequencing approaches usually take advantage of the linking information from paired-end reads of large-insert (e.g. fosmids) genomic libraries to create supercontigs (scaffolds). Please note that the term 'assembly' in the text subsequently refers to the assembly of contigs into scaffolds and not the assembly of reads into contigs, unless otherwise specified. In the absence of libraries, sequences of related organisms can also provide scaffolding information. Several software packages have been developed to handle this problem. Projector2 (1) provides a web interface to order prokaryotic assemblies by genome mapping. AMOScmp (2) applies a modified MUMmer algorithm to a newly sequenced genome by mapping it onto a reference genome. OSLay (3) is based on maximum weight matching to arrange the contigs of a target assembly, and this software also provides an interactive visualization of the computed layout. These methods are limited to the assembly of contigs derived from different strains of the same species, which may share nearly identical or highly conserved gene arrangements. For distinct species, however, assembly accuracy typically decreases dramatically.

One potential reason for reduced accuracy in contig assembly prediction is that currently used approaches cannot distinguish the optimal connections for contigs from a list of candidates. Currently employed methods usually use local information at each step, and the assembler can easily be confused by complex repeats or rearrangements, which leads to incorrect assemblies. In addition, previous methodologies also suffer from limitations with respect to reference genomes, because no gene arrangement information can provide correct and meaningful information to the assembly of a target genome. In the method described here, a novel scoring system was developed to evaluate the distance between two contigs in the target genome, and global search heuristics were used to predict the most probable pairwise connections for all contigs. Additionally, more than one genome was used as a reference to arrange the contigs. This can significantly improve the performance of predicting contig assembly for more distantly related genomes.

*To whom correspondence should be addressed. Tel: +1 814 865 1992; Fax: +1 814 863 7024; Email: dab14@psu.edu

Genetic algorithms (GAs) are effective stochastic global search algorithms inspired by the evolutionary features of biological systems (4). They start from a random population of individuals and iterate until some predefined stopping criterion is satisfied. Each individual is evaluated in each iteration (generation) based on its fitness function. GAs generally consist of three operators: selection, crossover and mutation. The selection operator is used to select the fittest individuals from the current population to serve as parents of the next generation. The crossover operator is used to generate offspring by exchanging genetic information between the two parents. The mutation operator randomly modifies individuals to prevent the search process from stagnation within a local optimum. After many generations, the population increases in average fitness and thus tends to give a better solution. Because they employ robust search methods that require little information to search efficiently in a large or poorly understood search space, GAs are particularly well-suited for solving complex optimization problems. In bioinformatics, they have been applied to align multiple sequences (5), to model genetic networks (6), to select regulatory structures (7), to analyze gene expression data (8) and to estimate phylogenetic inference (9) and recombination events (10). The ant colony optimization (ACO) metaheuristic method is a population-based approach that is used to solve combinatorial optimization problems (11). ACO mimics the way the real ants find the shortest route between a food source and their nest. Ants communicate with one another through pheromone trails and exchange information about which path should be followed. Such autocatalytic and positive feedback characteristics make ACO algorithms efficient methods for finding nearly optimal solutions to combinatorial optimization problems.

In this article, we describe a novel assembly algorithm (pheromone trail-based genetic algorithm, PGA), which consists of two major phases: calculation of the pairwise distance of the contigs to reference genomes and a global heuristic search for optimal contig connections. In the global heuristic search phase, we used a pheromone trail-based crossover to improve the performance of the GA. When generating offspring, it can obtain global information from pheromone trails, just as artificial ants do in ACO algorithms (12). Based on simulation studies, we demonstrate that PGA is a powerful and accurate method to assemble closely and moderately related genomes. We have successfully applied this algorithm to real incomplete genome data sets produced by Sanger DNA sequencing and pyrosequencing and compared the predictions of the program to the known assembly outcomes.

METHODS

Distance

Let A and B be a reference (complete) and a target (incomplete) genome, respectively, in which B is a collection of contigs $\{b_1, \ldots, b_n\}$. These contigs were then aligned to reference genome A using BLAST, and each contig b_i may have a set of matches $M_i = \{m_1, \ldots, m_r\}$, which results in $M = \{M_1, \ldots, M_n\}$. Each match m has two mapped positions along genome A (x_1, x_2). If two matches (m_i, m_j) overlap with $\ell_o > 0.2 \times \text{Min}\ \{\ell_i, \ell_j\}$, where ℓ_o, ℓ_i, ℓ_j are the lengths of the overlapping regions, m_i and m_j, respectively, the match with lower BLAST score will be removed from M. Likewise, the matches with $\ell < 200$ bp were also removed. After the removal of repeats and short matches, a revised $M^r = \{M_1^r, \ldots, M_n^r\}$ was then used to evaluate the distance between pairs of contigs.

For two given contigs b_i and b_j, their revised matching position sets are $M_i^r = \{m_{i1}, m_{i2}, \ldots, m_{ip}\}$ and $M_j^r = \{m_{j1}, m_{j2}, \ldots, m_{jq}\}$, respectively. Let $d_{i1,j1}, d_{i1,j2}, \ldots, d_{i1,jq}$ be the distance between m_{i1} and $m_{j1}, m_{j2}, \ldots, m_{jq}$, and $d_{i1,j} = \min\left\{d_{i1,j1}, d_{i1,j2}, \ldots, d_{i1,jq}\right\}$ be the minimal distance between m_{i1} and b_j. Similarly, the minimal distances between $m_{i2}, m_{i3}, \ldots, m_{ip}$ and b_j are $d_{i2,j}, d_{i3,j}, \ldots, d_{ip,j}$, and as such $D'_{i,j} = \left\{d_{i1,j}, d_{i1,j}, \ldots, d_{i1,j}\right\}$ represents the set of match distances between b_i and b_j. Different contigs, however, may have various cardinalities $|D'_{i,j}|$, which may bias the average distance between contigs. Hence, we only used the five smallest values in $D'_{i,j}$ to form a new set of match distances $D_{i,j}$. If $|D'_{i,j}| \leq 5$, we set $D_{i,j} = D'_{i,j}$. We then use a weighted method to evaluate each set of match distances D, which may possess five or fewer values. Let $\{x_1, x_2, \ldots, x_n\}$ represent the set of match distances, then the weighted-average distance $d^w = \sum_{i=1}^{n} x_i w_i$, where

$$w_i = \frac{\left[1/\left|x_i - 1/n\left(\sum_{i=1}^{n} x_i\right)\right|\right]}{\sum_{i=1}^{n}\left[1/\left|x_i - 1/n\left(\sum_{i=1}^{n} x_i\right)\right|\right]} (0 \leq n \leq 5).$$

We then use the rank of pairwise weighted-average distances to evaluate its fitness. For a given contig b_i, the weighted-average distance set between b_i and other contigs $b_1, b_2, \ldots, b_n$ is $D_i^w = \left\{d_{i1}^w, d_{i2}^w, \ldots, d_{in}^w\right\}$. $D_i^w$is divided into three subsets based on the actual distance, $D_i^{w1}(d^w \leq 8$ kb), $D_i^{w2}(8$ Kb $< d^w \leq 15$ kb), $D_i^{w3}(d^w > 15$ kb). The fitness between contig b_i and b_j is defined as follows:

$$f_{ij} = \begin{cases} |D_i^{w1}| & \text{if } d_{ij}^w \in D_i^{w1} \\ |D_i^{w1}| + |D_i^{w2}| & \text{if } d_{ij}^w \in D_i^{w2} \\ 10 & \text{if } |D_i^{w1}| = 0 \text{ and } j \leq 2 \\ 20 & \text{if } d_{ij}^w \in D_i^{w3} \text{ or } i = j \end{cases} \quad \mathbf{1}$$

$F_{B,A} = (f_{ij})_{n \times n}$ represents the fitness matrix of the contigs in genome B, when mapped onto the reference genome A. If two genomes were used as the reference at a time, $F_{B,A1A2} = (f'_{ij})_{n \times n}$, where

$$f'_{ij} = fi_{j1} \begin{cases} (f_{ij1} + f_{ij2})/2 & \text{if } f_{ij1}, f_{ij2} < 20 \\ f_{ij1} + 4 & \text{if } f_{ij1} < 20 \text{ and } f_{ij2} = 20 \\ f_{ij2} + 4 & \text{if } f_{ij2} < 20 \text{ and } f_{ij1} = 20 \\ 20 & \text{otherwise} \end{cases} \quad \mathbf{2}$$

If three or more reference genomes are used, a weighted combination of fitness matrix $F_{B,A1,\ldots,An}$ can be obtained in a similar way as shown in formula Equation (2).

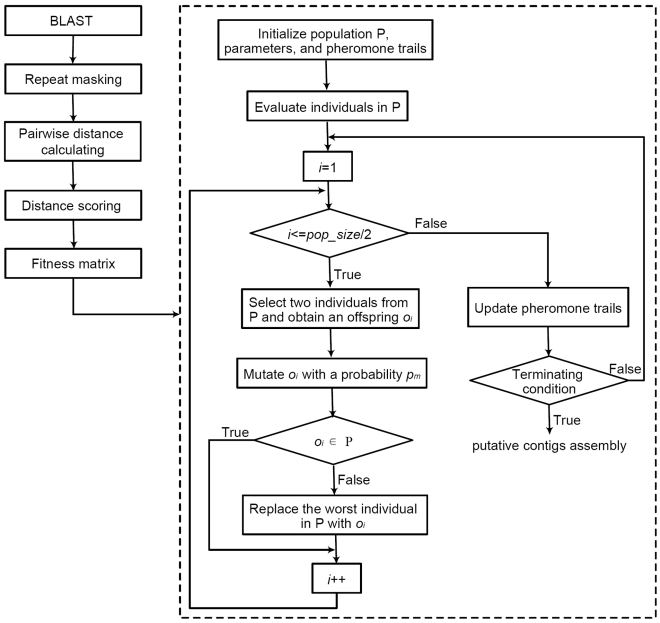

Figure 1. The flow chart of the pheromone trail-based genetic algorithm developed for genome assembly of contigs into scaffolds by comparison to one or more reference genome(s).

Pheromone trail-based genetic algorithm

After initialization of the parameters used in the algorithm, the GA begins by generating some feasible solutions (individuals) to constitute an initial population and then evaluating each of them with the objective (fitness) function. As long as the terminal condition is not satisfied, the GA is iterated. In each iteration (generation), individuals are probabilistically selected from the population according to the rule of roulette wheel selection. The selected individuals are then recombined and mutated to generate offspring. If a generated offspring is not the same as any individual in the population, it enters the population, and the worst individual in the population is removed to keep the population size constant. After the operators mentioned above, the pheromone trails are updated. A flow chart of the PGA is shown in Figure 1.

We use an integer string of length n as the representation of the chromosome (the possible connections of the contigs or solution), where n is the number of contigs. The fitness of a chromosome is derived from its solution quality. In the GA we use the rank of each chromosome in the population to evaluate its fitness (13). All chromosomes are first ordered according to their total lengths, and the chromosome with the minimum length gets rank 1. With these ranks, the fitness of a chromosome of rank i is calculated as follows: $fitness_i = \mathrm{Max} - [(\mathrm{Max} - \mathrm{Min})(i-1)/(k-1)]$, where k is the number of chromosomes in the population, Max is the maximum value of fitness, and Min is the minimum value of fitness. The best chromosome in the population gets fitness value Max and the worst chromosome gets fitness value Min. In our implementation, Max and Min are set to 1.2 and 0.8, respectively.

To generate the initial population, we first select a random contig as the first contig of an individual, and then choose the nearest contig (according to the distance mentioned in 2.1) from the contigs that have not appeared in the individual. This procedure is repeated until all contigs have appeared in the individual, and a feasible individual has been generated.

Of the three operators of a GA, crossover is well known to be the most important one because it significantly affects the performance of the GA. Previous crossover operators only took into account the position of the node or its order when generating offspring. This is generally considered 'blind' recombination (14). Here we implemented a new pheromone trail-based method, which utilizes both local (path length, adjacency relations) and global information to construct offspring. As shown in the Appendix (see Supplementary Data), the crossover operator employed attains global information from pheromone trails, as is done by artificial ants in the ACO algorithm (12). Our crossover operator was then tested under a pure GA framework and the computational results show that it gives a much faster and better convergence than order crossover-type operators.

Data application

The genome data used to evaluate the performance of our PGA and previous methods were downloaded from the National Center for Biotechnology Information (NCBI) or obtained from our ongoing genome projects in collaboration with the Joint Genome Institute (JGI; Department of Energy) or the Penn State Center for Comparative Genomics and Bioinformatics. Reference genomes were chosen based on phylogenetic proximity and the combined fitness score between them and the target genomes as calculated in the Distance section. The assembly accuracy is defined as $1 - N_b/N_g$, where N_b is the number of breakpoints identified when comparing two contig orderings (the correct one and the estimated one obtained by PGA or other approaches), N_g is the number of inter-contig gaps ($n-1$ in the case of linear chromosomes and n in the case of circular chromosomes). To make our explanation simple, we do not consider contig orientation in the following example. Let the real connection of five contigs be '1-2-3-4-5' and the estimated assembly be '1-2-4-3-5', then the accuracy of the estimated assembly is $2/4 = 0.5$. Hence, this criterion is quite stringent in evaluating the performance of contig assembly methods including PGA, Projector2, OSLay and other BLAST end-based tools.

The software can be accessed at http://sourceforge.net/projects/pga4genomics/. We provide both source code and pre-compiled versions of PGA implemented for Mac and PC. The following genome assemblies were run on a Power PC G5, Dual 2.3 GHZ, with 8GB SDRAM, and the computational time for heuristic search with PGA is usually several seconds. The time-consuming steps in assembling a genome with PGA are the BLAST and Repeat-masking as shown in Figure 1, which, depending on the genome size and the closeness between two genomes, usually takes from 5 to 30 min.

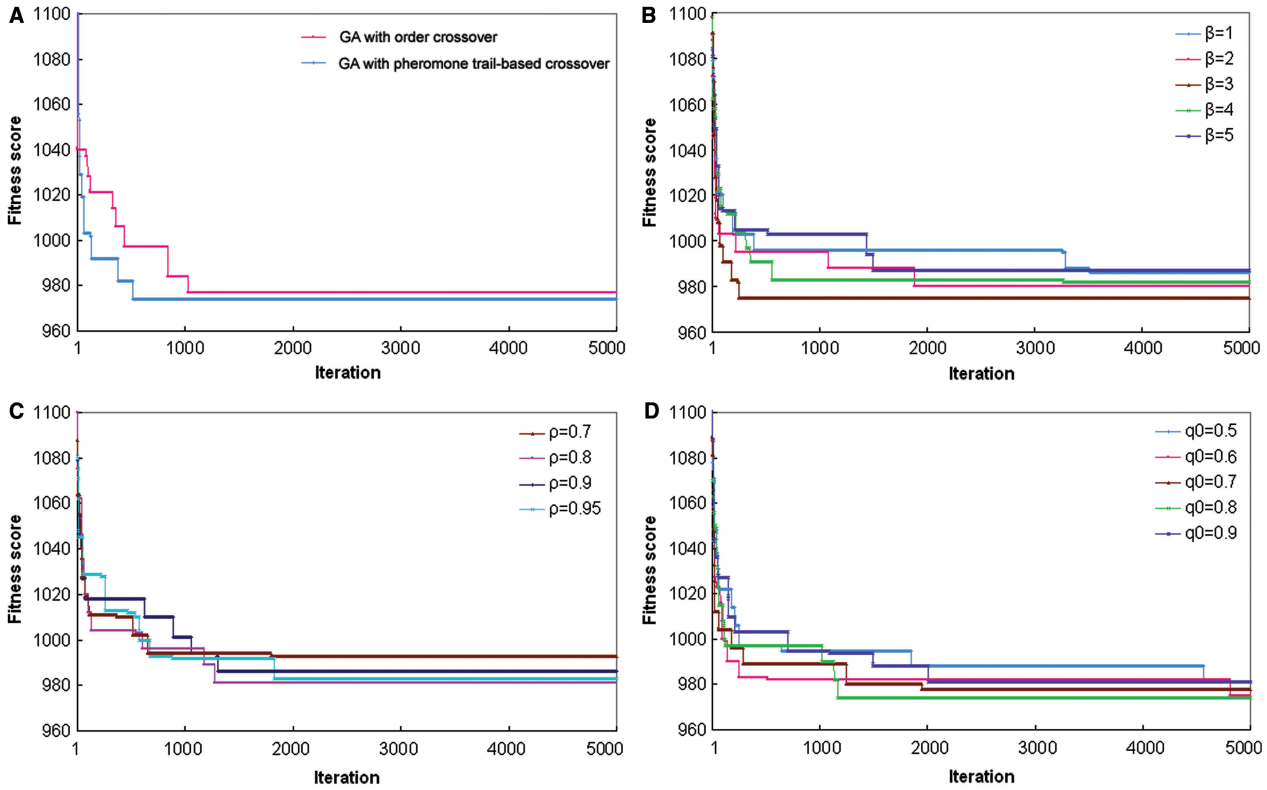

Figure 2. Relationships between PGA performance and various parameters. The *X*-axis shows the number of iterations, while the *Y*-axis shows the fitness score. (**A**) Comparison of the performance of genetic algorithms with different crossover operators. (**B–D**) Parameter settings for the relative importance of pheromone trail and the visibility (β), the pheromone trail persistence (ρ) and the probability for pseudo-random-proportional selection (q_0).

RESULTS

Parameters setting

The parameters considered here are those that have a direct effect on the pheromone trail-based crossover operator, including β, q_0 and ρ. Based upon working experience with PGA, the population size *pop_size* was set to 60, crossover and mutation probabilities to 1 and 0.1, respectively, and the number of iteration steps to 5000. The default values for β, q_0 and ρ were set to 3, 0.9 and 0.95, respectively. All parameter-setting tests were based on simulated data sets (120 contigs) produced by random shearing of the 4.97-Mb *Shewanella* sp. ANA-3 (Sana) genome and by using the 4.97-Mb genome of *Shewanella oneidensis* MR-1 (Sone) as the reference (15). As shown in Figure 2, at higher β-values, which mean local information (the visibility between contigs) earns a higher relative importance, quicker convergence can be found in the early evolutionary stages of the PGA. However, a higher β-value usually results in a more greedy selection in constructing offspring and causes a higher probability of early search stagnation. A higher q_0 in pseudo-random-proportional selection results in a quicker convergence at an earlier stage, and $q_0 = 0.8$ usually produces a better result here than other tested values. In Figure 2, we also present the evolutionary process under different ρ-values, which show that lower pheromone trail persistence may also cause earlier search stagnation.

To demonstrate the effectiveness of the pheromone trail-based crossover operator, we compared it to the well-known order crossover (16), and found that the pheromone trail-based crossover produced a much faster and better convergence (Figure 2A). As shown in Supplementary Figure S2, after 5000 iterations the average fitness score derived from the PGA is significantly smaller than that from the GA (Wilcoxon's signed-ranks test, $P = 1.76$ E-07). Given the random nature of any GA-based method, we compared the SD of the final optimized fitness score between the GA and the PGA, and found that the PGA is more stable and shows much smaller SDs (Wilcoxon's signed-ranks test, $P = 1.05$ E-03). Moreover, with an increasing number of contigs, both PGA and GA take more time to reach convergence; however, the actual computational CPU time for each run is only several seconds, so this is not a limitation. Although the optimal values for different data sets may be slightly different, in all subsequent analyses we set β, q_0, and ρ to 3, 0.8 and 0.8, respectively. In practice, this is not a limitation, because 5000 or 10 000 iterations are sufficient for the evolutionary process to reach convergence, and the final accuracy of the predicted assemblies does not seem to be affected significantly (data not shown).

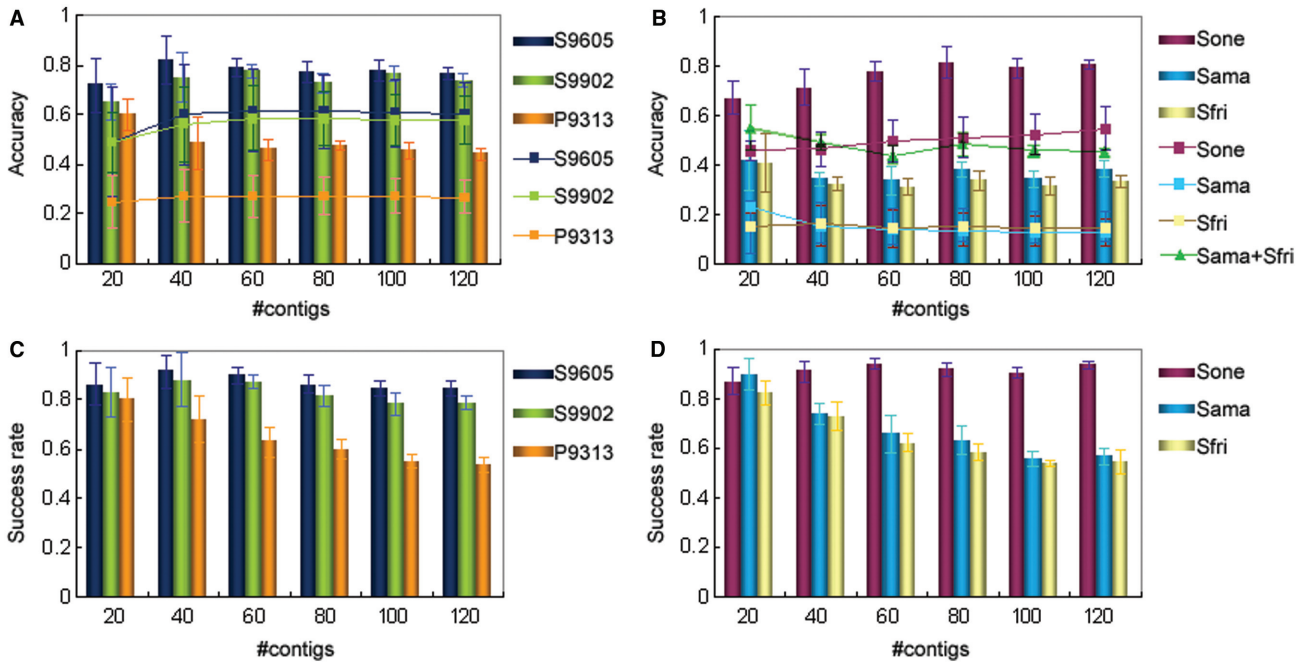

Figure 3. Performance comparison between the BLAST-end method (plotted points) and the PGA method (vertical bars) using different simulated data sets. (**A**) The assembly accuracies on data sets containing different numbers of simulated contigs of random fragments derived from *Synechococcus* sp. WH8102. The genomes of *Synechococcus* sp. CC9605 (S9605), *Synechococcus* sp. CC9902 (S9902) and *P. marinus* MIT 9313 (P9313) were used as the reference genomes, respectively. (**B**) The assembly accuracies on data sets containing different numbers of simulated contigs of random fragments derived from *Shewanella* sp. ANA-3. The genomes of *S. oneidiensis* (Sone), *S. amazonensis* (Sama), and *S. frigidimarina* (Sfri) were used as references, respectively. (**C**) The overall success rate for gap closure attained if one uses the four best predictions from the relaxed PGA method (PGA-extended, see text for detail) with the same data sets used in Figure 3A. (**D**) The overall success rate for gap closure attained using the four best predictions from the relaxed PGA method (PGA-extended, see text for detail) with the same data sets used in Figure 3B.

Simulation

We randomly generated 20, 40, 60, 80, 100 and 120 DNA fragments (i.e. simulated contigs) from a complete genome sequence. Each fragment was at least 5 kb, and the gap sizes between contigs ranged from −0.2 kb to 3 kb, where a negative value indicates that the two fragments actually overlapped. We repeated this process five times and thus generated five different sets of fragments for each category. Each fragment thus represents the equivalent of a contig derived from an actual draft genome assembly. We chose three other closely (or moderately) related genomes as references to predict the arrangement of these contigs. We then used these data sets to evaluate the efficiency of the PGA and to compare its performance with other known approaches. The most commonly used method to order and orient contigs is based on mapping contig ends to the reference genome (1,17–18). Because the aforementioned tools do not provide an automatic method to assemble large numbers of contigs, we modified them to produce easy-handling versions, designated collectively as BLAST-end methods, to benchmark the performance of our new algorithm.

We first used two strains of *Synechococcus* (*Synechococcus* sp. CC9605 (S9605); *Synechococcus* sp. CC9902 (S9902)) and one strain of *Prochlorococcus marinus* MIT9313 (P9313) (19) as reference genomes to assemble various numbers of pseudo-contigs simulated from *Synechococcus* sp. WH8102 (S8102) (20). These four genomes range from 2.2 to 2.5 Mb, and the synteny of these genomes is relatively well conserved. As shown in Figure 3A, PGA clearly outperforms the BLAST-end methods for all simulated data sets, and the overall performance of the PGA is quite similar when predicting the arrangements of different numbers of contigs. Similar results were also found using much larger *Shewanella* spp. genomes (4.3–5.0 Mb). Three species [*S. oneidensis*, (Sone), *S. amazonensis* (Sama) and *S. frigidimarina*, (Sfri)] were used to arrange the pseudo-contigs from *Shewanella* sp. ANA-3 (Sana). In contrast to Sone, Sama and Sfri have much less conserved gene syntenies with Sana, which results in assembly accuracies of <20% in most instances with the BLAST-end methods (Figure 3B). We also determined the accuracy of predicted contig arrangements by using the PGA with Sama, Sfri and a combination of Sama and Sfri as the reference genomes. As shown in Figure 3B, the results indicate that the predicted organization of the Sana genome, assembled by the PGA, has significantly higher accuracy (40–50%) than that assembled by the BLAST-end methods (<20%). Additionally, the data in Figure 3B show that using a combination of two reference genomes can significantly improve the contig arrangement predicted by PGA. It is notable that the assembly accuracy by PGA did not decrease significantly as the number of simulated contigs increased. This strongly supports the idea that PGA would be especially useful for projects with lower sequence coverage and greater numbers of contigs [e.g. draft genome sequences with only 2× coverage (21)]. To verify

Table 1. Comparisons of contig assembly accuracy for authentic data sets from the assembly of contigs from green sulfur bacterial genomes using PGA, PGA-extended, BLAST-end, Projector2 and OSLay

Genomes[a]	Contigs	Method	Reference Ctep		Reference Cpha		Reference Plut		2 or 3 Refs	
			Best	Average	Best	Average	Best	Average	Best	Average
Clim	37	PGA	0.324	0.276 ± 0.040	0.405	0.373 ± 0.032	0.378	0.346 ± 0.026	0.514[b]	0.443 ± 0.040[b]
		PGA-ext.[e]	0.514	NA	0.541	NA	0.568	NA	NA	Na
		BLAST-end	0.108	NA	0.135	NA	0.135	NA	NA	NA
		Projector2	0.189	NA	0.189	NA	0.162	NA	NA	NA
		OSLay	0.135	NA	0.162	NA	0.108	NA	NA	NA
Cvib	26	PGA	0.385	0.331 ± 0.031	0.462	0.454 ± 0.015	0.769	0.738 ± 0.015	0.731[c]	0.731 ± 0.000[c]
		PGA-ext.[e]	0.577	NA	0.731	NA	0.885	NA	NA	NA
		BLAST-end	0.115	NA	0.385	NA	0.538	NA	NA	NA
		Projector2	0.231	NA	0.308	NA	0.577	NA	NA	NA
		OSLay	0.000	NA	0.154	NA	0.423	NA	NA	NA
Cpar	58	PGA	0.690	0.679 ± 0.014	0.431	0.400 ± 0.020	0.586	0.559 ± 0.018	0.741[d]	0.738 ± 0.007[d]
		PGA-ext.[e]	0.914	NA	0.621	NA	0.724	NA	NA	NA
		BLAST-end	0.534	NA	0.190	NA	0.172	NA	NA	NA
		Projector2	0.224	NA	0.121	NA	0.155	NA	NA	NA
		OSLay	0.534	NA	0.052	NA	0.103	NA	NA	NA

[a]Assembly of contigs of *C. limicola* (Clim), *C. vibrioforme* (Cvib) and *C. parvum* (Cpar) contigs.
[b]*Chlorobium tepidum* (Ctep), *C. phaeobacteriodes* (Cpha) and *P. luteolum* (Plut) were used as the reference genomes.
[c]*Chlorobium phaeobacteriodes* (Cpha) and *P. luteolum* (Plut) were used as the reference genomes.
[d]*Chlorobium tepidum* (Ctep) and *P. luteolum* (Plut) were used as the reference genomes.
[e]The corresponding value indicates the overall success rate for gap closure attained using the four best predictions from the PGA-extended method.
NA, not applicable.

that the PGA outperforms the actual methods (Projector2, OSLay) instead of the proxy method (BLAST-end), we used simulated data sets from Sana to evaluate their performance (Supplementary Figure S3). The assembly results show that when using closely related species (Sone) as a reference genome, both PGA and OSLay give significantly higher assembly accuracies than Projector2 and BLAST-end (Wilcoxon's signed-ranks test, $P \ll 0.001$), but that there was no significant difference between PGA and OSLay (Wilcoxon's signed-ranks test, $P = 0.793$). In contrast, when using moderately related species as references, PGA outperforms all other algorithms including OSLay, Projector2 and BLAST-end (Wilcoxon's signed-ranks test, $p \ll 0.001$).

The accuracy of the predicted assembly fully depends on the conservation of gene synteny between the target and reference genomes, and thus poorly conserved gene order may provide insufficient, confusing or incorrect information for arranging the target contigs. To account for these possibilities and uncertainties, PGA predicts three additional suboptimal connections for each contig from a fitness matrix ($F_{B,A}$). As shown in Figure 3C and D, by relaxing the selection criteria for optimal connections one can increase the probability of predicting the correct contig organization significantly, especially for less-conserved genomes. In practical terms, these predictions can be used to reduce significantly the number of PCR reactions required to obtain products to close gaps between contigs. The practical outcome is that closing costs and time can be substantially reduced.

Evaluation on real data sets

To evaluate the performance of PGA further, we assembled three green sulfur bacterial (GSB) genomes [*Chlorobium limicola* (Clim), *C. vibrioforme* (Cvib) and *Chlorobaculum parvum* (Cpar)] by using three previously finished GSB reference genomes. Each of these genomes has now been completely closed, so it is possible to evaluate the prediction of PGA relative to the known, final contig organization in each genome (Zhao, F., Li, T. and Bryant, D.A., unpublished results). The first two genomes were sequenced using the standard Sanger techniques, and the last one was sequenced by pyrosequencing with a Roche GS20 instrument (22). After initial assembly of reads into contigs (by Phred/Phrap/Consed for the first two genomes and the Newbler assembler for *C. parvum*), we obtained a set of contigs for each genome, and then removed those contigs with lengths <3.5 kb. As shown in Table 1, we used *C. tepdium* (Ctep), *C. phaeobacteroides* DSM 266 (Cpha), *Pelodictyon luteolum* DSM 273 (Plut) as reference genomes, some of which have limited synteny with the target genome. In all three real data sets, the PGA outperformed any other assembly method (BLAST-end, Projector, and OSLay). For Clim, when we used Ctep, Cpha and Plut as references simultaneously, the assembly accuracy was significantly improved to 51.4%. In contrast, the average assembly accuracy for BLAST-end, Projector2 and OSLay was only 13.5, 18.9 and 16.2%, respectively. It is worth noting that the combination of several genomes as the reference may not assure a better result, especially when selecting very closely related and moderately related genomes together, because the latter may actually obscure connections among contigs. For example, the assembly accuracy was 76.9% for Cvib when using Plut as a reference, whereas assembly accuracy decreased slightly (73.1%) when a second reference genome (Cpha) was included. When *C. parvum* (Cpar) was sequenced and the data from the GS20 pyrosequencer

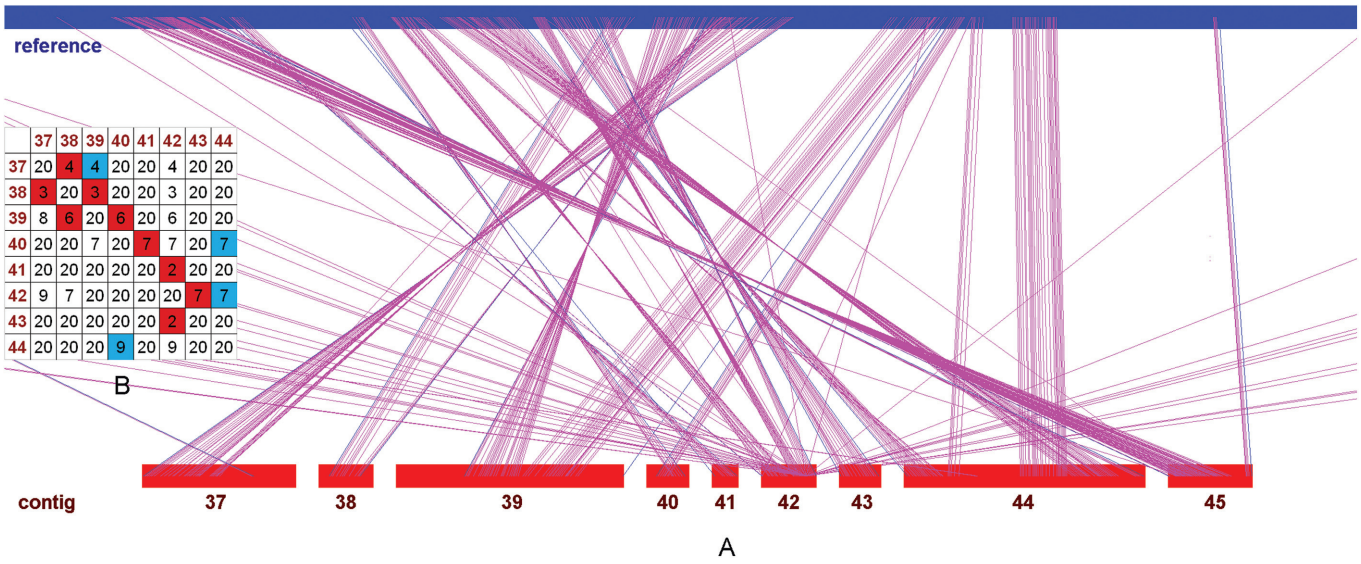

Figure 4. An example to illustrate the performance of the PGA. (**A**) Linear mapping between given target contigs (red bars) and a reference genome (blue bar). The purple lines represent the connections of the interior BLASTN matches and the light blue lines represent the connections of the terminal contig BLASTN matches. The name (number) of each contig is indicated below the red bars. (**B**, inset) The fitness matrix for these contigs derived from the reference genome. The smaller the matrix value, the shorter the distance between a pair of contigs. Red boxes indicate scores that provide correct information to PGA; the blue boxes indicate scores that provide false information, which may predict an incorrect assembly.

was initially assembled into contigs, there were more contigs than for the other two species because of the shorter pyrosequencing read lengths (~120 bp). PGA dramatically outperformed any of the other assembly tools listed in Table 1, especially when using more distantly related species as reference genomes (Cpha or Plut).

The PGA program can provide additional suboptimal connections for each contig. This significantly increases the range of predictions and can provide one with many more clues as to how to close gaps by PCR, especially for distantly related genomes. For example, by including three additional, suboptimal connections in PCR-based testing of connectivities, one can substantially increase the probability of performing the appropriate PCR reaction without having to test all possible reactions. PGA has been successfully applied to several currently ongoing genome sequencing projects in collaboration with the Penn State Center for Comparative Genomics and Bioinformatics or the Joint Genome Institute. The genome sizes of these organisms range from about 2 Mb to 6 Mb (*H. pylori* J166a, *H. pylori* J166b, *H. pylori* Mom, *H. cetorum*, *Bordetella bronchiseptica* 1289, *Chloroflexus aurantiacus* Y-400-fl, and *C. parvum*) and most of these genomes have been sequenced by pyrosequencing without any paired read sequencing. PGA has greatly facilitated the closing and finishing phases of these projects.

DISCUSSION

We have developed a GA-based algorithm (PGA) for comparative genome assembly of contigs into scaffolds from Sanger sequencing or pyrosequencing data using one or multiple reference genomes. Comparisons with previously published algorithms indicate that PGA can successfully obtain optimal or nearly optimal solutions from complex and large numbers of possible solutions. Our algorithm outperforms previous approaches for the following reasons: (i) PGA applies a novel scoring system to evaluate the distance between two contigs. This system is more reliable and informative than the simple linear arrangement produced for the target and reference genomes, as employed by other tools. (ii) PGA uses global search heuristics to find the optimal connection for each contig from a collection of possible candidates, and thus avoids misguidance from regions with less-conserved gene synteny. (iii) PGA can use multiple genomes as references to arrange the contigs into scaffolds. This significantly improves performance when assembling distantly related genomes.

Figure 4 illustrates why our proposed algorithm outperforms other approaches. Here we selected the mapping results between eight simulated contigs of *Shewanella* sp. ANA-3 and the reference genome (*S. oneidensis* MR-1). Red lines between the target and reference genomes show the positions and connections of BLAST matches, and blue lines show the terminal significant matches (*E*-value $<1E$-10) with length $\geq$200 bp. Apparently, terminal matches for each contig do not assure the right connection to their adjacent contig. For example, the right end of contig 37 was targeted to another far away region in the reference, and likewise there are also many uninformative connections between the target and the reference genome, which result from non-orthologous BLAST matches and/or gene rearrangement. Using the BLAST-end method, only contig 42 and contig 43 were correctly ordered but with the wrong orientation. However, the PGA, which utilized a novel distance-scoring matrix and performed a global search of optimal connections, gave a much better prediction. For example, contig 38 has three candidate

connections (contig 37, contig 39, contig 42 in line 2, Figure 4B), but for contig 37, contig 39 and contig 42 (in row 1/3/6, Figure 4B), their closest contigs are contig 38, contig 38, and contig 41, respectively. The scores labeled in red in this partial matrix may be successfully searched by PGA and thus give correct predictions, whereas the scores labeled in blue may give wrong predictions. In this case, all of these contigs were correctly ordered except contig 44. In contrast, if we use other methods, such as MUMmer, Projector2 and OSLay, all of these contigs would be misassembled because of frequent gene inversion and rearrangements among them.

In summary, the new PGA contig assembler differs considerably from other assembly tools and has crucial advantages over them. Even for difficult assembly problems and using reference genomes with less overall conserved synteny, PGA achieves much more accurate assemblies than other tools by using a global search strategy and by introducing more than one reference at a time. Additionally, the extended PGA can also predict suboptimal connections for each contig and thus can increase the chances of closing a gap by predicting a set of the most probable contig connectivities. Extensive testing conducted so far indicates that it is extremely useful for finishing bacterial genomes. The main drawback of this comparative assembly method is its dependence on the availability of reference sequences of related genomes. However, with the very rapid growth in the number of sequenced genomes, and especially the introduction of newly developed sequencing and resequencing methodologies (e.g. pyrosequencing), PGA should have an increasing number of applications.

SUPPLEMENTARY DATA

Supplementary Data are available at NAR Online.

ACKNOWLEDGEMENTS

We thank Dr Ji Qi for help with the application of the algorithm to some ongoing genome sequencing projects at Penn State. We are grateful to two anonymous referees and to Dr Richard Alvey for comments and assistance in improving the manuscript. This study was supported by NSF grant MCB-0523100 to D.A.B. Funding to pay the Open Access publication charges for this article was provided by MCB-0523100.

Conflict of interest statement. None declared.

REFERENCES

1. van Hijum,S.A., Zomer,A.L., Kuipers,O.P. and Kok,J. (2005) Projector 2: contig mapping for efficient gap-closure of prokaryotic genome sequence assemblies. *Nucleic Acids Res.*, **33**, W560–W566.
2. Pop,M., Phillippy,A., Delcher,A.L. and Salzberg,S.L. (2004) Comparative genome assembly. *Brief. Bioinform.*, **5**, 237–248.
3. Richter,D.C., Schuster,S.C. and Huson,D.H. (2007) OSLay: optimal syntenic layout of unfinished assemblies. *Bioinformatics*, **23**, 1573–1579.
4. Holland,J.H. (1992) Genetic algorithms. *Sci. Amer.*, **4**, 44–50.
5. Nguyen,H.D., Yoshihara,I., Yamamori,K. and Yasunaga,M. (2002) Aligning multiple protein sequences by parallel hybrid genetic algorithm. *Gen. Inform.*, **13**, 123–132.
6. Kikuchi,S., Tominaga,D., Arita,M., Takahashi,K. and Tomita,M. (2003) Dynamic modeling of genetic networks using genetic algorithm and S-system. *Bioinformatics*, **19**, 643–650.
7. Gilman,A. and Ross,J. (1995) Genetic-algorithm selection of a regulatory structure that directs flux in a simple metabolic model. *Biophy. J.*, **69**, 1321–1333.
8. Ooi,C.H. and Tan,P. (2003) Genetic algorithms applied to multi-class prediction for the analysis of gene expression data. *Bioinformatics*, **19**, 37–44.
9. Lewis,P.O. (1998) A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data. *Mol. Biol. Evol.*, **15**, 277–283.
10. Kosakovsky Pond,S.L., Posada,D., Gravenor,M.B., Woelk,C.H. and Frost,S.D. (2006) Automated phylogenetic detection of recombination using a genetic algorithm. *Mol. Biol. Evol.*, **23**, 1891–1901.
11. Syswerda,G. (1991) A study of reproduction in generational and steady-state genetic algorithms. In *Foundations of Genetic Algorithms*. Morgan Kauffmann Publishers. San Mateo, USA, pp. 94–101.
12. Shtovba,S.D. (2005) Ant algorithms: theory and applications. *Program. Comp. Soft.*, **4**, 167–178.
13. Whitley,D. (1989) The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best. *Proceedings of the Third International Conference on Genetic Algorithms*, 116–121.
14. Dorigoa,M. and Blum,C. (2005) Ant colony optimization theory: a survey. *Theor. Comp. Sci.*, **344**, 243–278.
15. Heidelberg,J.F., Paulsen,I.T., Nelson,K.E., Gaidos,E.J., Nelson,W.C., Read,T.D., Eisen,J.A., Seshadri,R., Ward,N., Methe,B. *et al.* (2002) Genome sequence of the dissimilatory metal ion-reducing bacterium *Shewanella oneidensis*. *Nat. Biotechnol.*, **20**, 1118–1123.
16. Stakweather,T., McDaniel,S., Mathias,K., Whitley,C. and Whitley,D. (1991) A comparison of genetic sequencing operators. In *Proceedings of the 4th International Conference on Genetic Algorithms*. Morgan Kauffmann Publishers, San Mateo, USA, pp. 69–76.
17. Yu,Z., Li,T., Zhao,J. and Luo,J. (2002) PGAAS: a prokaryotic genome assembly assistant system. *Bioinformatics*, **18**, 661–665.
18. Frangeul,L., Glaser,P., Rusniok,C., Buchrieser,C., Duchaud,E., Dehoux,P. and Kunst,F. (2004) CAAT-Box, contigs-assembly and annotation tool-box for genome sequencing projects. *Bioinformatics*, **20**, 790–797.
19. Rocap,G., Larimer,F.W., Lamerdin,J., Malfatti,S., Chain,P., Ahlgren,N.A., Arellano,A., Coleman,M., Hauser,L., Hess,W.R. *et al.* (2003) Genome divergence in two *Prochlorococcus* ecotypes reflects oceanic niche differentiation. *Nature*, **424**, 1042–1047.
20. Palenik,B., Brahamsha,B., Larimer,F.W., Land,M., Hauser,L., Chain,P., Lamerdin,J., Regala,W., Allen,E.E., McCarren,J. *et al.* (2003) The genome of a motile marine *Synechococcus*. *Nature*, **424**, 1037–1042.
21. Green,P. (2007) 2 x genomes — does depth matter? *Genome Res.*, **17**, 1547–1549.
22. Margulies,M., Egholm,M., Altman,W.E., Attiya,S., Bader,J.S., Bemben,L.A., Berka,J., Braverman,M.S., Chen,Y.J., Chen,Z. *et al.* (2005) Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, **437**, 376–380.