MDAsim: a Multiple Displacement Amplification Simulator

Zeinab Taghavi
Department of Computer Science
Wayne State University
Detroit, MI 48202
ztaghavi@wayne.edu

Sorin Draghici
Department of Clinical and Translational Science
Department of Computer Science
Wayne State University
Detroit, MI 48202
sorin@wayne.edu

***Abstract*—Multiple displacement amplification (MDA) is a fast non-PCR based isothermal DNA amplification method that can amplify small amounts of DNA samples to a reasonable quantity for genomic analysis. This technique is suitable for metagenomics and single cell genome sequencing and related analyses. The distribution of the coverage of the output amplicons for single cell MDA unlike the multicell amplification is not uniform. This distribution is unknown, and the parameters that affect this amplification bias have not been studied thoroughly. To have a better understanding of the MDA reaction we have developed a simplified mathematical model and the corresponding simulation algorithm called MDAsim to obtain a generative model for the output amplicons. In this paper we will show that the output coverage of MDAsim matches the experimental coverage very well. Therefore, the combination of MDAsim and a sequencing simulator can be utilized for test and evaluation of single cell assemblers avoiding the burden of experimental sequencing. Our results suggest that modelling the MDA mechanism and simulation of such increasingly complex models may provide valuable insight into the MDA process, which in turn can be used to design more efficient MDA reactions.**

***Keywords*-DNA sequencing; Multiple displacement amplification (MDA); Single cell amplification; DNA Amplification simulator.**

I. Introduction

Multiple displacement amplification (MDA) is a non-PCR based DNA amplification method that can amplify small amounts of DNA samples to a reasonable quantity for genomic analysis [1]. MDA is a fast isothermal reaction and compared to other amplification methods generates less amplification bias [1]. It also generates larger amplicons with a lower error rate. These characteristics make this technique suitable for metagenomics and single cell genome sequencing and related analyses. MDA has been used to amplify DNA as small as a plasmid and as large as a whole mammalian genome [1], [2], [3]. In particular, Chitsaz *et al.* have recently demonstrated that high-quality draft assemblies of bacterial genomes can be obtained from single cells using MDA [3]. The vast majority of environmental bacteria, for instance in and on human body, live in symbiotic environments, which makes them uncultivable outside of their natural habitat. Single cell genomics based on MDA enables study of uncultivable bacteria [3]. These advances put MDA in the center of attention for single cell genomic and transcriptomic studies.

Although MDA is a powerful amplification method, it introduces errors, biases, and artifacts during the amplification process, like all other amplification methods. The reflection of these procedural characteristics in the final product affects the accuracy of downstream analysis, e.g. sequence assembly, alignment, variation detection, etc., and therefore, requires modelling and mitigation *in silico*. With the advent of low-cost high-throughput DNA sequencing, it is now possible to quantify MDA products in a more systematic and comprehensive way [3]. There are two plausible frameworks to do so:

1) treat MDA as a blackbox, collect large datasets from various MDA experiments, and try to construct non-mechanistic models for the observed input-output data,
2) construct increasingly complex biochemical/mechanistic models for the MDA reaction, and try to obtain a generative model for the output data.

The latter is desirable also from a different viewpoint. We would like to obtain a detailed picture of the effect of various variables, for instance the initial concentration of primers and enzymes, on the MDA process to improve the technique itself. Therefore by modelling and simulating the MDA process, not only will we be able to model and mitigate the output artifacts and biases but also we may be able to design improved experimental MDA reactions. In this paper, we offer MDAsim, an MDA simulator software, and evaluate it by comparing the coverage distribution of MDAsim output with that of actual MDA. To the best of our knowledge, this is the first approach toward building a simulator for the MDA process. Additionally, our simulator can be used to try MDA *in silico* before committing to costly and time-consuming experimental wet lab efforts.

II. Method

A. Simulated Amplification Process

The MDA is an isothermal reaction that uses the DNA polymerase from bacteriophage $\Phi29$ and random primers to amplify a DNA strand. The DNA strand first gets denatured. During the entire period of the process, primers attach randomly to single-stranded areas of a tree-like branching structure and the $\Phi 29$ polymerase synthesizes new strands from the older templates. When the polymerase reaches a double stranded region, it displaces the previously extended strand and continues to synthesize its own strand [1]. The result of this reaction will be a tree of single- and double-stranded DNA in a complex branching structure. The bacteriophage $\Phi 29$ has the capacity to perform DNA synthesis for more than 70,000 nucleotides (nt) without dissociating from the template [1] and is stable enough to efficiently synthesize DNA for many hours [1]. At the end of the process, S1 nuclease is applied to cleave single stranded fragments from double stranded parts and dismantle the entire tree-like structure.

B. Model Assumptions and Notations

We implemented the reaction assuming that the time is discretized into equal intervals. In the beginning of the process, at time 0, we assume one or several fragments of single-stranded DNA sequences and limited number of available primers and $\Phi 29$ are put in the solution. To amplify a double-stranded DNA, the two reverse-complement single-stranded DNA sequences will be input. The primers bind randomly to available single-stranded regions and $\Phi 29$ polymerases instantly attach to the primed locations and synthesize the complementary strand. The length of the new strand is a random number which has the average of $m_{\Phi 29}$ nt and standard deviation of $\sigma_{\Phi 29}$ nt. Based on the law of mass action, we assume that the rate of reaction of primers binding is proportional to the product of the concentrations of the available primers in the solution and available single stranded fragments. Therefore, we calculate the number of primers bound at time step t as

$$n_b(t) = n_b(0)\frac{l_s(t)}{l_s(0)}\frac{n_p(t)}{n_p(0)} = \alpha l_s(t) n_p(t),$$

where $l_s(t)$ is the length of single-stranded fragments and $n_p(t)$ is the number of available primers in the solution at time t. The parameter $\alpha = n_b(0)/[l_s(0)n_p(0)]$ is the input of the software which is the number of primers bound at time 0 normalized to the available length and available primers. The parameter α can be interpreted as defining the speed of the binding reaction.

Another parameter given as the input to the software is k, the number of nucleotides that can be synthesized by a single $\Phi 29$ in a single time interval. This parameter also defines the speed of the second part of the reaction which is the synthesis of new amplicons. The process stops when there are no available primers or no available single strand fragments to copy new fragments from, or the coverage reaches a given number C.

III. Algorithm

The inputs of the simulator are DNA strand(s), the list of primers, both in fasta format, and parameters α, k, $n_p(0)$, $m_{\Phi 29}$, $\sigma_{\Phi 29}$, C. In our software we assumed that all of the primers have equal size. This assumption is based on the use of the popular random hexamer primers [1].

There are several main variables used in the program. The list *availablePrimerCounts* keeps track of the number of each primer still available in the solution. The variable *fragmentList* is a list of all synthesized fragments. To keep track of single- and double-stranded regions of each fragment, for each base we store whether it is released or attached to a fragment at what position. Displacement of a strand is modelled by change of status form occupied to released of the corresponding bases in the fragment. We have another list named *activePhi29List*. This list contains fragments to which $\Phi 29$ polymerases are attached. For these fragments the process of synthesis is not finished yet. To keep track of available positions in the *fragmentList* that a primer can attach to, we index single stranded positions in a list called *availablePrimerPositionsList*. To bind a new primer, a position is selected randomly from this list. The available positions should be indexed every time a strand branches from its template and is added to the list. Also positions which become unavailable in any stage of the reaction should be removed from the list. Besides, if a number of available primers in the solution are finished, i.e., the corresponding term in *availablePrimerCounts* is zero, all of the corresponding available positions should be removed from the *availablePrimerPositionsList*. Construction of this variable is the main challenge in our implementation, since this list should be capable of having random access while it should be easy to add to or remove from.

In the end, fragments are cleaved from the positions where a single strand meets the double stranded region. The output of the software is a list of amplicons written in the fasta format. The output file name is given as the input. In summary, the implemented algorithm is as follows:

1) $t = 0$: Read the input DNA strand(s), the input primers, and parameters α, k, $n_p(0)$, $m_{\Phi 29}$, $\sigma_{\Phi 29}$, and C. The DNA strand is initialized as the first fragment in the fragment list.
2) Iterate until the average coverage reaches the desired coverage, C, or no new binding of primers is possible.
    a) $t = t + 1$.
    b) Calculate $n_b(t)$. Select $n_b(t)$ random available positions from *availablePrimerPositionsList* and attach the corresponding primers to the templates. Then
        i) For each attached primer, create new fragments in the *fragmentList*.
        ii) Add the new fragments to the *activePhi29List*.
        iii) Update the *availablePrimerPositionsList* by removing the newly occupied position in the list.

iv) For each attached primer, reduce the primer counts in the *availablePrimerCounts*.

v) Remove those primers whose counts in *availablePrimerCounts* are zero and their corresponding available positions from the list *availablePrimerPositionsList*.

c) For each fragment in the *activePhi29List*, add k new complemented nucleotides copied from their template strand. In this section, some fragments may be displaced by newly synthesized fragments. Then

i) For the fragments that have achieved their desired length, detach the $\Phi29$ and remove the fragment from the *activePhi29List*.

ii) The desired length of each synthesized fragment is a random number with uniform distribution with mean $m_{\Phi 29}$ nt and standard variation of $\sigma_{\Phi 29}$ nt.

iii) Update the *availablePrimerPositionsList* by removing the newly occupied positions from the list, or adding the newly released regions to the list.

3) Cleave all of the fragments from the positions where the double- and single-starnded regions meet.

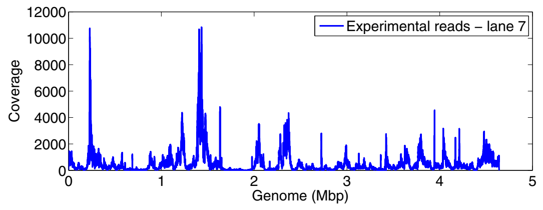
4) Write the amplicons in the output file in fasta format. The intermediary lists *availablePrimerPositionsList* and *fragmentList* are saved in two files for debugging purposes.

IV. Results

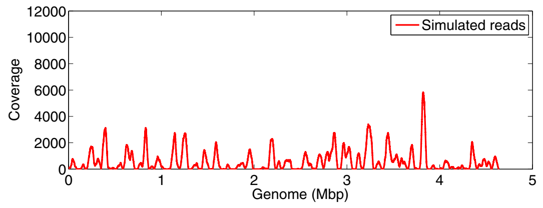
We have used MDAsim to simulate amplification of single cell Staphylococcus aureus and Escherichia coli. We applied an Illumina read generator, ART [4], to generate Illumina reads from the synthesized amplicons. We compared our results with the experimental outputs from the single cell Illumina reads of [3] available in `http://bix.ucsd.edu/singlecell/`.

The process of generating the single cell data is described in [3] in details. Three single cells (2 E. coli and 1 S. aureus) are picked from microbial samples. Each molecule which its size is in the order of femto gram is amplified to reach around 1 microgram DNA material. The output is sequenced by Illumina GAIIx sequencer to reach the average of 600x and 1800x coverage for E. Coli and S. aureus, respectively.

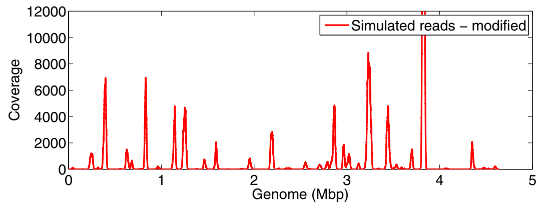
To generate the E. coli amplicons with the coverage of 600x, we ran MDAsim with the following parameters: $\alpha = 2.6 \times 10^{-11}$, $k = 10000$, $n_p(0) = 250000$, $m_{\Phi 29} = 80000$, $\sigma_{\Phi 29} = 2300$, and $C = 600$. For S. aureus DNA sequence, we ran the software with the following parameters to have amplicons with the coverage of 1800x: $\alpha = 3.6 \times 10^{-11}$, $k = 10000$, $n_p(0) = 400000$, $m_{\Phi 29} = 80000$, $\sigma_{\Phi 29} = 2300$, and $C = 2000$. The output amplicons were feed to the software ART to generate the output reads. The coverage of the results are shown is Figures 1 and 2.



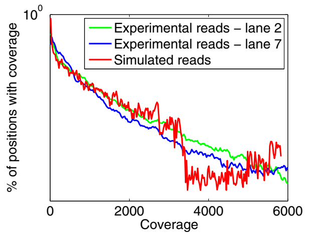
(a)



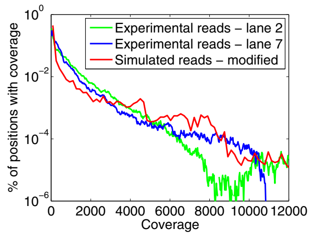
(b)



(c)



(d)



(e)

Figure 1. Comparison of the coverages of E. coli experimental reads, lane 2 and 7, and simulated reads generated by MDAsim. Figures (a)-(c) are illustrating coverage per genome positions for experimental lane 7 and simulated reads. The coverage of the modified simulated reads is generated by raising the coverage presented in (b) to the power of 2.9, to match 10^9x amplification gain of MDA. Figures (d) and (e) show the corresponding log-scaled coverage histograms for graphs (b) and (c), respectively. Long tails on the x-axis of Figures (d)-(e) has been clipped to show the details of the graphs.

In the real world MDA, the amplification gain is in the order of 10^9 to amplify a small DNA of size femto gram to a reasonable volume of microgram. This volume is down sampled to 600x (1800x) coverage in the sequencer. Simulating a billion fold coverage for MDA is not tractable with the current version of the simulator. To approximate 10^9x coverage, we can assume that we apply the 600x (1800x) amplification process for l consecutive times. In fact, if $c_{600x}(n)$ is the coverage of the nth nucleotide after 600x amplification, i.e., $\sum_{n=1}^{N} \frac{1}{N} c_{600x}(n) = 600$, the $c_{10^9x}(n)$ can be approximated by $c_{600x}(n)^l$. In this case, l

can be estimated such that $\sum_{n=1}^{N} \frac{1}{N} c_{600x}(n)^l \approx 10^9$. For the data given in Figure 1, $l \approx 2.9$ and for Figure 2, $l \approx 2.7$. To generate the actual reads with the modified coverage distribution, a greedy approach can be used in which existing fragments and subfragments are replicated a sufficient number of times. We leave this part for future work.

To compare the output of our simulator with the experimental result, we have used coverage histogram plots as comparison criteria. Software BWA version 0.6.1 [5] was used for alignment of the reads to the reference genome. In the histogram plot, x axis is the basewise coverage and y axis is the percentage of the number of loci with x coverage in the genome. It is shown that this measure can distinguish between normal multicell coverage and single cell MDA coverage distributions [3]. For the multicell sequencing, the histogram has normal distribution with the mean equal to the average coverage. For the single cell, the coverage frequency spreads along a wider area and reduces as the coverage increases.

Figure 1 compares the coverage of the reads from E. coli experimental results and simulated results generated by the MDAsim and ART software. Out of the 7 lanes of E. coli experimental results, there are only two independent instances of MDA amplified cells. Here we have chosen lanes 2 and 7 as the two independent representatives. Figures 1a-1b are the plots of the coverage per genome loci for lane 7 and simulated result. Figure 1c is the modified simulated coverage gained by raising the coverage presented in 1b to the power of 2.9 to match 10^9x amplification gain of MDA. Comparison of the log-scaled histograms in Figures 1d-1e show that the distribution of the simulated result matches the experimental result very well for coverage less than 3250. However, for coverage above 3250 the simulated distribution drops rapidly because of the difference in amplification gain in experiment and simulation. As can be seen in 1e, after applying the approximation method the distribution of the coverage is matching the experimental result very well for the coverages between 2000 and 12000.

Figure 2 illustrates the comparison of S. aureus coverage histograms for experimental and simulated reads. As can be seen the distributions have the same behavior for simulation and experimental result for coverages less than 6000. For the modified approximated coverage which is the simulated read coverage raised to the power of 2.7 the distributions have good agreements between 4000 to 10000 coverage.

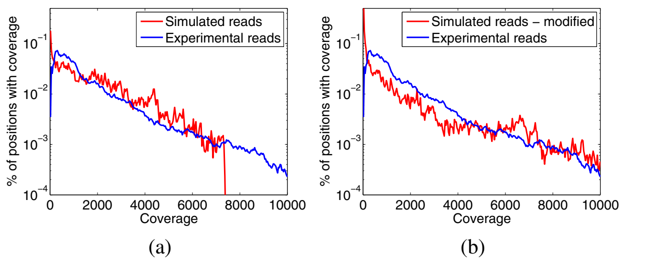


Figure 2. Comparison of the log-scaled coverage histograms of S. aureus experimental reads and simulated reads generated by MDAsim. The coverage of the modified simulated reads is generated by raising the coverage presented in (a) to the power of 2.7, to match 10^9x amplification gain of MDA.

V. Conclusion

We developed a novel simulator MDAsim to model the MDA process. The results show that in those applications for which high coverage peaks are not important, MDAsim output provides a good approximation of the read coverage. For the remaining extremely high coverage portion, we gave a simple and efficient post-processing correction algorithm. We conclude that MDAsim combined with a sequencing simulator, e.g., ART [4] for Illumina, can be used for test and evaluation of single cell assemblers without the burden of experimental sequencing.

Our results suggest that modelling and simulation of the MDA mechanism by increasingly complex models in the future may provide valuable insight into the MDA process, which in turn can be used to design more efficient MDA reactions. This is the first attempt toward simulating MDA and is expected to gradually evolve toward a more accurate model including incorporation of substitution and chimeric errors. In addition, the next version of the simulator needs to be parallelized to increase its speed.

Availability: The MDAsim software is available upon request.

Acknowledgment

This work was supported by the grants NIH 5R01DK089167-02, NIH 2R42GM087013-02, and NSF DBI-0965741.

References

[1] F. B. Dean, J. R. Nelson, T. L. Giesler, and R. S. Lasken, "Rapid amplification of plasmid and phage DNA using Phi 29 DNA polymerase and multiply-primed rolling circle amplification," *Genome Res.*, vol. 11, pp. 1095–1099, Jun 2001.

[2] S. Hosono, A. F. Faruqi, F. B. Dean *et al.*, "Unbiased whole-genome amplification directly from clinical samples," *Genome Res.*, vol. 13, pp. 954–964, May 2003.

[3] H. Chitsaz, J. L. Yee-Greenbaum, G. Tesler *et al.*, "Efficient de novo assembly of single-cell bacterial genomes from short-read data sets," *Nature Biotech*, vol. 29, no. 10, pp. 915–921, Oct 2011.

[4] W. Huang, L. Li, J. R. Myers, and G. T. Marth, "ART: a next-generation sequencing read simulator," *Bioinformatics*, vol. 28, no. 4, pp. 593–594, Feb 2012.

[5] H. Li and R. Durbin, "Fast and accurate short read alignment with Burrows-Wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, Jul 2009.