Fundamental Research 4 (2024) 785–794

Contents lists available at ScienceDirect

Fundamental Research

journal homepage: http://www.keaipublishing.com/en/journals/fundamental-research/

Article

Generating barcodes for nanopore sequencing data with PRO

Ting Yu[a,1], Zitong Ren[a,1], Xin Gao[b,*], Guojun Li[a,*], Renmin Han[a,*]

[a] Research Center for Mathematics and Interdisciplinary Sciences, Frontiers Science Center for Nonlinear Expectations (Ministry of Education), Shandong University, Shandong 266000, China

[b] Computer, Electrical and Mathematical Sciences and Engineering Division & Computational Bioscience Research Center, King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia

ARTICLE INFO

Article history:
Received 30 June 2023
Received in revised form 20 February 2024
Accepted 9 April 2024
Available online 25 April 2024

Keywords:
Third-generation sequencing
Nanopore sequencing
DNA barcode
Farthest point sampling algorithm
High throughput

ABSTRACT

DNA barcodes, short and unique DNA sequences, play a crucial role in sample identification when processing many samples simultaneously, which helps reduce experimental costs. Nevertheless, the low quality of long-read sequencing makes it difficult to identify barcodes accurately, which poses significant challenges for the design of barcodes for large numbers of samples in a single sequencing run. Here, we present a comprehensive study of the generation of barcodes and develop a tool, PRO, that can be used for selecting optimal barcode sets and demultiplexing. We formulate the barcode design problem as a combinatorial problem and prove that finding the optimal largest barcode set in a given DNA sequence space in which all sequences have the same length is theoretically NP-complete. For practical applications, we developed the novel method PRO by introducing the probability divergence between two DNA sequences to expand the capacity of barcode kits while ensuring demultiplexing accuracy. Specifically, the maximum size of the barcode kits designed by PRO is 2,292, which keeps the length of barcodes the same as that of the official ones used by Oxford Nanopore Technologies (ONT). We validated the performance of PRO on a simulated nanopore dataset with high error rates. The demultiplexing accuracy of PRO reached 98.29% for a barcode kit of size 2,922, 4.31% higher than that of Guppy, the official demultiplexing tool. When the size of the barcode kit generated by PRO is the same as the official size provided by ONT, both tools show superior and comparable demultiplexing accuracy.

1. Introduction

Pooling multiple samples to maximize the utilization of resources is a common practice in biomolecular research. To identify different samples, experimental platforms generally link barcodes (short and unique nucleotide sequences) to different samples. Barcodes play an important role in research and biotechnology, such as single-cell sequencing [1], gene synthesis [2], drug discovery [3], and high-throughput sequencing [4-9]. However, DNA synthesis and sequencing errors that are manifested as insertions, deletions, or substitutions of bases can corrupt barcodes, preventing them from being correctly identified and reducing the accuracy of experiments. To improve accuracy, researchers have usually employed error-correcting code schemes to design DNA barcodes, which are used mainly in computer science for error detection and correction in information transmission. Hamady et al. used the Hamming code, a well-studied error correction code in computer science, to design barcodes. However, this approach does not take insertion or deletion errors into account [5]. To address this challenge, the Levenshtein code was introduced to design barcodes that can address all kinds of sequencing errors, but the length of each corrupted barcode must be known [6,7,10]. The FREE barcodes developed by Hawkins et al. can handle editing errors even when the corrupted barcode length is unknown [8]. Wang et al. presented the TVNS algorithm to generate barcodes and improve the lower bound of the size of the barcode sets [9].

Nanopore sequencing is a breakthrough in modern sequencing technology that has greatly promoted studies of genome assembly [11], single-cell transcriptomics [12-14], and fusion genes [15] due to its advantage in long sequencing reads. To reduce sequencing costs, multiple samples are generally sequenced simultaneously by linking a unique barcode for each individual sample. The sequencing reads for each sample are then separated by the known barcodes. For instance, Tian et al. used equal mixes of five cancer cell lines (~40 cells per line) profiled with matched Illumina and Nanopore reads to study the comprehensive characterization of single-cell full-length isoforms [16]; Xiaoying Fan et al. efficiently separated single-cell information according to the 24-nucleotide (nt) Nanopore platform cell barcodes [17]. The throughput of nanopore sequencing is lower than that of next-generation short-read sequencing, which introduces high cost and complex operations in

* Corresponding authors.
E-mail addresses: Xin.gao@kaust.edu.sa (X. Gao), guojunsdu@gmail.com (G. Li), hanrenmin@gmail.com (R. Han).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.fmre.2024.04.014

research [16-19]. However, the official barcode kits provided by ONT contain at most 96 barcodes, which means that at most 96 samples can be sequenced in a single run. Therefore, to sequence many samples simultaneously, sufficient barcodes are needed. Achieving higher-throughput nanopore sequencing requires the support of barcode kits with larger sizes. To our knowledge, current barcode design methods were specifically developed for next-generation sequencing with a low error rate, as the greatest drawback of nanopore sequencing is the high error rate [12]. Therefore, designing barcodes for nanopore sequencing is worth investigating.

Here, we provide a comprehensive study of the barcode design problem and demultiplexing strategy from both theoretical and practical perspectives. To tolerate sequencing errors and ensure correct identification after sequencing, barcodes should be sufficiently separated from each other in a barcode kit. During sequencing, barcodes are subject to errors with a certain probability. From a probabilistic perspective, we innovatively define a new measure of divergence between two short DNA sequences, namely, probability divergence, and demonstrate that probability divergence is more accurate than the most widely used distance measure in demultiplexing, edit distance. We formulate the barcode generation problem as a combinatorial problem and prove that finding the optimal largest barcode set in a given space of short DNA sequences of the same length is NP-complete. From a practical application point of view, we developed the software package PRO (https://github.com/rztongr/PRO), which can generate and demultiplex barcodes. The main idea behind the proposed method is the farthest-point sampling algorithm, which ensures that any two barcode sequences in the generated barcode set are sufficiently distinct from each other. According to our simulation experiments, PRO significantly expands the capacity of DNA barcode kits. PRO designs a barcode kit of size 2,292, the same length as the official barcode, with an achieved demultiplexing accuracy of 98.29%. When the barcode kit we designed was the same size as the officially available kit, PRO performed comparably to the official demultiplexing tool Guppy in terms of demultiplexing accuracy.

2. Methods

To maximize the utilization of resources in research, DNA barcoding is a powerful tool for sample identification. Barcode kits with large sizes and demultiplexing strategies with high accuracy make the experiments efficient and accurate. For this purpose, we developed an open-source package PRO, which can be used for generating and demultiplexing barcodes. To generate an optimal barcode set, PRO first preprocesses the candidate sequences according to user requirements and introduces a novel metric (i.e., probability divergence) to measure the similarity between sequences; it then employs the farthest-point sampling algorithm to select the barcodes, which ensures that the generated sequences are adequately distinguishable from each other. To accurately demultiplex corrupted barcodes, PRO uses flanking sequences to extract the corrupted barcodes from the DNA sequences and then identifies the barcodes according to the probability divergence.

2.1. Preliminary knowledge

Denote $\Sigma = \{A, G, C, T\}$ as a 4-arq alphabet. A seq of length k over alphabet Σ is a sequence $s_1 s_2 \cdots s_k$, $s_i \in \Sigma$, $i = 1, 2, \cdots, k$. We define the k-mer space as the set of all seq of length k, denoted by S_k. The union of all k-mer spaces is denoted by S; *i.e.*, $S = U_{k>0} S_k$.

For accurate demultiplexing, any two sequences in a barcode set should be sufficiently distinguishable at a certain distance metric, denoted by dis. Given a barcode length n and threshold t, we aim to find a subset of S_n, denoted by $\{barcode\}$, in which the dis between any two sequences is greater than or equal to t. The goal of generating a barcode set with a maximum cardinality of length n is to select as many distinct sequences as possible from among the 4^n sequences in S_n. In brief, the barcode sets to be generated should satisfy

$$dis(seq', seq'') \geq t, \ for \ \forall seq', seq'' \in \{barcode\},$$

where seq' *and* seq'' are two barcode sequences and $\{barcode\}_{max}$ is the barcode set with maximum size. Now, we prove that generating $\{barcode\}_{max}$ is a NP-complete problem.

The maximum clique problem (MCP), which is to find a complete graph with the maximum vertices in an undirected graph, is a classical combinatorial optimization problem that is NP-complete [20]. Let $G = (V, E)$ be an undirected graph, V be the vertex set of G, and $E \in V \times V$ be the edge set of G. A graph $G = (V, E)$ is complete if all its vertices are pairwise adjacent. A subgraph U of G is a clique of G if and only if U is complete and is not contained in a larger complete subgraph of G. The maximum clique problem requires a clique of maximum cardinality in an undirected graph $G = (V, E)$.

Proposition 1: Given sequence length n and threshold t, finding the maximum barcode set $\{barcode\}_{max}$ in S_n is a NP-complete problem.

Proof: The problem of finding the maximum barcode set $\{barcode\}_{max}$ can be transformed into the MCP. First, we define the weight w(u, v) between two vertices u, v as w(u, v) = dis(u, v). We construct the complete graph $G' = (V', E')$ with each of the 4^n sequences in S_n as a vertex. The weight of the edge between any two vertices u, v in G' is w(u, v). Given the threshold t, we define $G = (V, E)$, where $V = V'$ and $E = \{(u, v) \in G' | w(u, v) \geq t\}$.

On the other hand, finding the maximum barcode set involves finding a maximum set of sequences in which each pair of sequences is adequately distinguishable, e.g., the distance metric between each pair of sequences is adequately high. Thus, we consider a threshold t and suppose that a distance metric between two sequences that is greater than t is sufficient for demultiplexing. Then, finding the maximum barcode set in S_n is the problem of finding the maximum clique in graph G, which means that finding the maximum barcode set $\{barcode\}_{max}$ in S_n is an NP-complete problem.

After sequencing, one DNA sequence may become another sequence with arbitrary length and arbitrary base composition. We now prove that the sum of the probabilities of a sequence becoming any other sequence is 1.

Proposition 2: Denote $p(seq, seq')$ as the probability of seq transforming into seq'. For all $seq \in S$,

$$\sum_{seq' \in S} p(seq, seq') = 1$$

Proof: Given $seq = s_1 s_2 \cdots s_n \in S$, where $S = U_{n>0} S_n$ and $s_i \in \{A, G, C, T\}, i = 1, 2 \cdots n$, denote seq' as the sequence after sequencing; as mentioned, the length of seq' is arbitrary. Suppose the numbers of matches, deletions, and substitutions in the transformation from seq into seq' are m, d, and s, respectively. It is obvious that $m + s + d = n$. Denote P_m, P_s, *and* P_d as the probabilities of the occurrence of matching, deleting, and substituting; then, we have $P_m + P_s + P_d = 1$.

Therefore, the probability of m insertions, d deletions and s substitutions occurring in $s_1 s_2 \cdots s_n$ during sequencing is

$$\sum_{m+s+d=n} C_n^s P_s^{\ s} C_{n-s}^d P_d^{\ d} C_{n-s-d}^m P_m^{\ m}$$
$$= \sum_{m+s+d=n} \frac{n!}{s!d!m!} P_s^{\ s} P_d^{\ d} P_m^{\ m} = \left(P_m + P_s + P_d\right)^n = 1$$

For the insertions, there are $n + 1$ intervals in the sequence $s_1 s_2 \cdots s_n$. Suppose that the probability of insertions occurring in an interval is $P_{i'}$ (note that multiple bases may be inserted) and that the probability of no insertions occurring is $P_{ni'}$. Denote i' and ni' as the numbers of intervals where insertions occur and insertions do not occur, respectively; then,

$$P_{i'} + P_{ni'} = 1$$

$$i' + ni' = n + 1$$

Similarly,

$$\sum_{i'+ni'=n+1} C_t^{i'} P_{i'}{}^{i'} C_{n-t}^{ni'} P_{ni'}{}^{ni'} = \left(P_{i'} + P_{ni'}\right)^{n+1} = 1$$

Therefore, considering all kinds of cases,

$$\sum_{m+s+d=n} C_n^s P_s{}^s C_{n-s}^d P_d{}^d C_{n-s-d}^m P_m{}^m \sum_{i'+ni'=n+1} C_t^{i'} P_{i'}{}^{i'} C_{n-t}^{ni'} P_{ni'}{}^{ni'}$$

$$\left(P_m + P_d + P_s\right)^n \left(P_{i'} + P_{ni'}\right)^{n+1} = 1.$$

Denote k as the total number of inserted bases. The sequence $s_1 s_2 \cdots s_n$ is transformed into seq' after m, s, d, k matches, substitutions, deletions, and insertions, where $m+s+d=n$, $i'+ni'=n+1$ and seq' may be any sequence in S.

2.2. Overview of the probability divergence

Errors occur during nanopore sequencing in the form of insertions, deletions and substitutions of DNA bases. After sequencing, the bases and lengths of the barcodes generally change, and they may become arbitrary DNA sequences. Suppose that the probabilities of a base being correctly sequenced, deleted, substituted and inserted during sequencing are known; then, for any two DNA sequences seq, seq', where seq is corrupted into seq' during sequencing, the probability of seq becoming seq' can be computed theoretically. Defining the divergence between two DNA sequences based on these probabilities is worth investigation.

2.2.1. Definition of probability divergence

There are various ways for a sequence seq to become seq'. For instance, Fig. 1a shows three ways in which $seq = AGTC$ becomes $seq' = AGGT$. Obviously, the number of ways for seq to become seq' is infinite. However, for many of these ways, such as way_3, their probability of occurring is actually very low during sequencing, so they make only a small contribution to the calculation of the probability of $AGTC$ becoming $AGGT$. Therefore, the probability of transformations such as way_3 is not considered in this research.

Edit distance is an effective metric for measuring the degree of difference between two sequences; it is defined as the minimum number of edit operations (insertion, deletion, substitution) required to transform one sequence into another. We calculate the edit distance via a dynamic programming algorithm and obtain the alignment matrix. The alignment of the two sequences can be obtained by backtracking in the alignment matrix, but this alignment may not be unique. For example, $edit\_distance(AGTC, AGGT) = 2$ in Fig. 1a, but way_1 and way_2 are two different alignments.

Denote p_m, p_d, p_s, p_i as the probabilities of a single base being correctly sequenced, deleted, substituted, and inserted in nanopore sequencing, respectively. This research is based on two assumptions: the occurrences of insertions, deletions, and substitutions for each base are independent during sequencing, and $p_d = p_s = p_i = \frac{1}{3}(1-p_m)$. Therefore, the probabilities of insertion, deletion, and substitution for a specific base $N \in \{A, G, C, T\}$ are $\frac{1}{4}p_i$, p_d, and $\frac{1}{3}p_s$; note that $p_d > \frac{1}{3}p_s > \frac{1}{4}p_i$ under these assumptions. When backtracking in the alignment matrix, deletion is considered first, followed by substitution and then insertion, which gives a unique alignment result $Align$. Suppose the numbers of base matches, insertions, deletions, and substitutions in $Align$ are m, i, d, s (Table 1). We define the divergence between two sequences from a probabilistic perspective based on the edit distance.

Table 1
Detailed information for calculating the probability divergence.

	Match	Deletion	Substitution	Insertion
p	p_m	p_d	$\frac{1}{3}p_s$	$\frac{1}{4}p_i$
$number$	m	d	s	i

For any two DNA sequences X, Y, there is a unique alignment $Align$ of X to Y, and the probability divergence of X with respect to Y is defined as

$$ProDiv(X,Y) = p_m{}^m p_d{}^d (\tfrac{1}{3}p_s)^s (\tfrac{1}{4}p_i)^i.$$

For example, for the sequences $AGTC$ and $AGGT$, based on the alignment Way_1, the probability divergence of $AGTC$ with respect to $AGGT$ is $ProDiv(AGTC, AGGT) = \frac{1}{4}p_m{}^3 p_d p_i$.

2.2.2. The properties of probability divergence

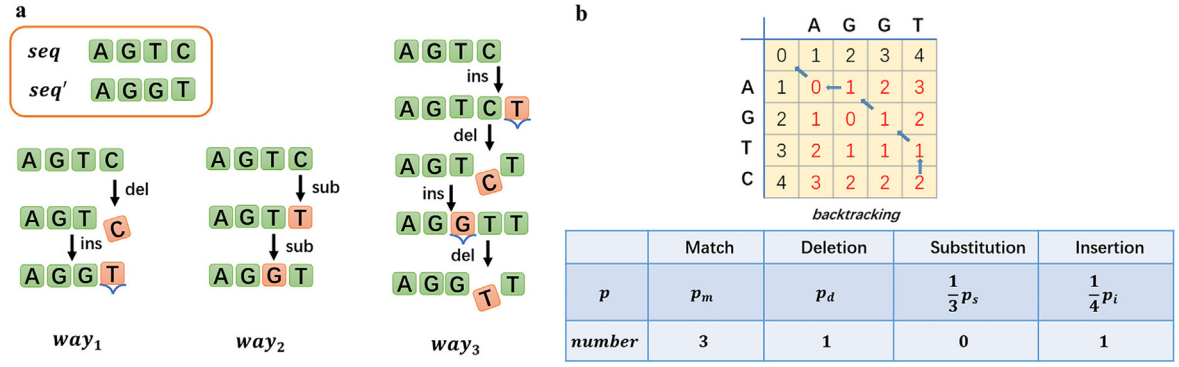
According to the definition of probability divergence, the more similar two sequences are, the greater the probability of divergence between the two sequences. However, probability divergence is not a distance metric since it does not even satisfy symmetry; e.g., $ProDiv(AGGT, AGTC) = \frac{1}{9}p_m{}^2 p_s{}^2$ but $ProDiv(AGTC, AGGT) = \frac{1}{4}p_m{}^3 p_d p_i$. Here, we define the weight between two sequences X, Y as $weight(X,Y) = max\{ProDiv(X,Y), ProDiv(Y,X)\}$.

The probability divergence proposed in the research is a rough measure of the probability that one sequence will transform into another since, as mentioned above, some ways for a sequence to change to another sequence are not considered. Therefore, the sum of the probability divergences calculated in this research is less than 1. For instance, when $n = 5$ and the sequencing error rate is 0.12, for any $seq \in S_5$, we calculate the sum of the probability divergence between seq and all the sequences in S_4, S_5 and S_6, and most of them are greater than 0.85.

2.3. Barcode generation strategy

2.3.1. Generation of barcodes

We proved that finding $\{barcode\}_{max}$ is an NP-complete problem by transforming the $\{barcode\}_{max}$ generation problem into the



	Match	Deletion	Substitution	Insertion
p	p_m	p_d	$\frac{1}{3}p_s$	$\frac{1}{4}p_i$
$number$	3	1	0	1

Fig. 1. A simple example illustrating the definition of probability divergence. (a) Three ways of editing $AGTC$ to $AGGT$. Edit distance ($AGTC$, $AGGT$)=2, and way_1 and way_2 show two backtracking paths from AGTC to AGGT, the steps of which are minimal. way_3 shows that a substitution is equivalent to an insertion and a deletion, but way_3 is complex. (b) An example showing how to find the unique alignment $Align$ from $AGTC$ to $AGGT$ when defining probability divergence by backtracking. The table displays the numbers of matches, insertions, deletions, and substitutions in the unique alignment.

maximum clique problem; therefore, a polynomial-time algorithm cannot be found. Here, we use farthest-point sampling (FPS), a heuristic algorithm, to generate the barcode set. Specifically, the candidate sequences are regarded as sampling points in a space, and the distance between each pair of sampling points is defined as the weight between the corresponding sequences. The core idea of the algorithm is to iteratively select the farthest point from the space such that any two points in the sampled set, denoted by $\{barcode\}_{set}$, are as far apart as possible; in other words, the probability divergence is as small as possible. For a given dataset, denoted by $\{candidate\}$, and the number of points that should be sampled, denoted by n, we present the procedure for FPS in Algorithm 1. To cope with different sequencing error rates, we adapt the algorithm by setting the corresponding threshold t accordingly, and the algorithm terminates if $weight(\{barcode\}_{set}, seq) > t$, *i.e.*, $max\{weight(s, seq), s \in \{barcode\}_{set}\} > t$, which ensures that the maximum probability divergence between any two sequences in $\{barcode\}_{set}$ is less than or equal to t. Each time a sequence seq1 is selected, we remove the sequence s satisfying $ProDiv(s, seq1) > t$ in $\{candidate\}\backslash\{barcode\}_{set}$ to reduce the number of calculations.

Algorithm 1

Input: $\{candidate\}$, sample number n
Output: $\{barcode\}_{set}$
Function FPS ($\{candidate\}$, n) begin
Randomly select a sequence seq
$\{barcode\}_{set} \leftarrow \{seq\}$
$\{candidate\} \leftarrow \{candidate\}\backslash\{seq\}$
While $|\{barcode\}_{set}| \leq n$:
  select the sequence seq in $\{candidate\}$ farthest from $\{barcode\}_{set}$
  $\{candidate\}=\{candidate\}\backslash\{seq\}$
$\{barcode\}_{set} \leftarrow \{barcode\}_{set}\{seq\}$
End while

Algorithm 2 shows the adapted farthest-point sampling algorithm. To optimize the sampling step, we use an array of size *n*, where each location in the array stores the distance between a point and the last sampled sequence *seq* and *n* is the size of the input {candidate}. When sampling the next sequence, it is sufficient to calculate the *n* distances and update the array. In addition, we reduce the computational effort by reducing the candidate sequence with a set threshold *t*.

Algorithm 2

Input: $\{candidate\}$, threshold t
Output: $\{barcode\}_{set}$
Function FPS Adapted $\{candidate\}$, t) begin
Randomly select a sequence seq
$\{barcode\}_{set} \leftarrow \{seq\}$
$\{candidate\} \leftarrow \{candidate\}\backslash\{seq\}$
While $weight(\{barcode\}_{set}, seq) \leq t$ do :
  for s $\in$ $\{candidate\}$:
    if $weight(s, seq) > t$:
    $\{candidate\} \leftarrow \{candidate\}\backslash\{s\}$
  end for
  select seq that satisfies
$weight(seq, \{barcode\}_{set}) = \min\{weight(seq, \{barcode\}_{set}), seq \in \{candidate\}\}$
  $\{candidate\} = \{candidate\}\text{-}\{seq\}$
  $\{barcode\}_{set} \leftarrow \{barcode\}_{set}\{seq\}$
End while

2.3.2. Determination of the threshold

The threshold *t* determines the maximum probability divergence between any two barcode sequences in the generated $\{barcode\}$. A smaller threshold indicates that the difference between two sequences in the generated $\{barcode\}$ is greater, and the size of $\{barcode\}$ is smaller; thus, the demultiplexing accuracy is expected to be greater. Assuming that the length of the barcodes is *n* and the sequencing error rate is *p*, we set the threshold *t* by converting the number of errors that are expected to be correctly demultiplexed into the probability divergence. Under the aforementioned assumption $p_d = p_s = p_i$, if the threshold *t* ensures that the barcodes with fewer than *k* sequencing errors will be correctly demultiplexed, there are at most 4^k cases; among these, the probability of those in which all the *k* errors are insertions is the lowest, denoted by $p' = (1-p)^n(\frac{1}{4}p_i)^k$, and the threshold *t* is set to $\sqrt{p'}$. However, it is conservative to determine the threshold in this way since the barcode set generated by the farthest-point sampling algorithm can be correctly demultiplexed in the case of more than *k* errors.

2.3.3. Dataset preselection

Given *n*, the length of the barcode, there are up to 4^n candidate DNA sequences. For example, the official DNA barcode length given by ONT is 24 bp, indicating that the barcode set should be generated from $4^{24} = 281,474,976,710,656$ candidate sequences. Therefore, to improve the efficiency of barcode generation, we preselect the 4^n DNA sequences to effectively reduce the number of candidate sequences. The preselected set is denoted as $\{candidate\}_{pre}$. Specifically, each candidate DNA sequence with length n is hashed into a 64-bit integer with hashing function N: $N(G)=0$, $N(A)=1$, $N(T)=2$, $N(C)=3$ such that for sequence $s_1s_2\cdots s_n \in S_n$, $N(s_1s_2\ldots s_n) = N(s_1)4^0 + N(s_2)4^1 + \ldots + N(s_n)4^{n-1}$. Since the size of the numbers is a somewhat reliable indicator that the corresponding sequences are different, we preprocess the data in the following way: Given the initial dataset $\{candidate\}$ and preselected size $|\{candidate\}_{pre}|$, we first hash the sequence in $\{candidate\}$ into a 64-bit integer set $\{num\}$. The integers in $\{num\}$ are sorted in descending order and then equally divided into $|\{candidate\}_{pre}|$ sets. Finally, by picking a random integer from each set, $\{candidate\}_{pre}$ consists of the DNA sequences corresponding to the $|\{candidate\}_{pre}|$ integers according to the hash function.

2.4. Demultiplexing strategy

In nanopore sequencing, DNA samples first require library preparation to convert them into a format suitable for Oxford Nanopore Technology Devices, after which the barcode is placed between two known flanking sequences (Fig. 2a). As shown in Fig. 2b, our demultiplexing strategy consists of two steps: (1) extracting the corrupted barcode s' from the DNA sequence after sequencing and (2) demultiplexing based on s'. For corrupted barcode s', s_0 is regarded as the demultiplexing sequence if $pro(s', s_0) = \max\{pro(s, s'), s \in \{barcode\}\}$ since s_0 is the barcode with the highest probability of being corrupted to s' in $\{barcode\}$. We call this demultiplexing strategy PRO-dex.

As mentioned above, the barcode is placed between two known flanking sequences. Denote the corrupted DNA sequence after sequencing as $sequence'$. The two known flanking sequences are aligned to $sequence'$ using the alignment tool edlib [21], and the sequence between the two flanking sequences is the corrupted barcode we extract. Then, we demultiplex the corrupted barcode using PRO-dex. Fig. 3a illustrates scenarios where barcodes are not extracted accurately in a simulated experiment, which results in more errors in the extracted barcodes than would theoretically be caused by sequencing, culminating in incorrect demultiplexing. Thus, precisely extracting the corrupted barcode s' in the first step can substantially improve the accuracy of demultiplexing.

The rear top and the rear bottom flanking sequences are aligned to $sequence'$ by edlib to obtain the minimum edit distance D_f, D_b; then, the start and end positions of the two flanking sequences on $sequence'$ can be obtained, denoted by $start_f$, end_f, $start_b$, and end_b (Fig. 2b). To extract the barcode sequences more accurately, we make an approximate judgment about the position of the flanking sequences in $sequence'$. When extracting barcodes, instead of aligning the flanking sequences directly to $sequence'$, we narrow the range of sequences to those that align with $sequence'[35{:}77]$ and $sequence'[35 + start_f{:}120]$, where the integers are the index values on $sequence'$. Fig. 3c1 and c2 show the results of not narrowing and narrowing the range of sequences used to align, which clearly indicates that the latter is more accurate than the former.

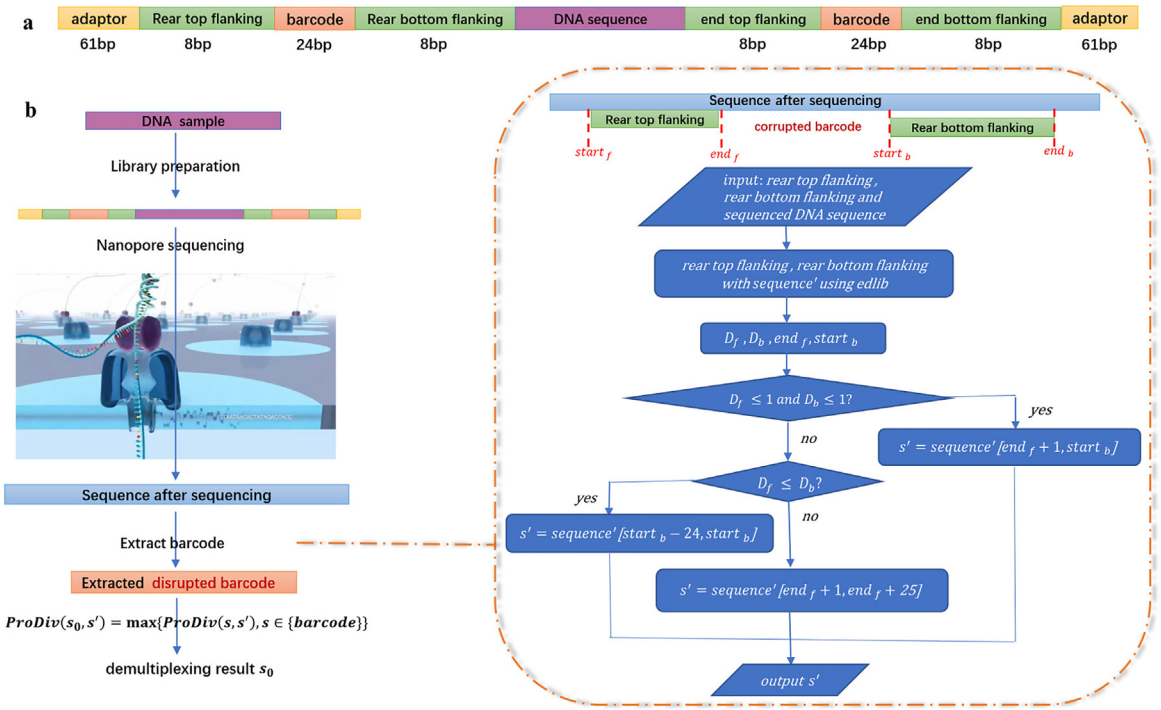


Fig. 2. The demultiplexing strategy. (a) Schematic of the DNA sequence structure used for nanopore sequencing devices. DNA sequences are prepared by attaching adapters, flanking sequences, and DNA barcodes to strand ends using either ligation-based or rapid chemical methods. (b) The demultiplexing strategy. We first extract the barcode by aligning the rear top and bottom flanking regions with $sequence'[35{:}77]$ and $sequence'[end_f{+}1, 120]$ using edlib to obtain positions end_f,$start_b$. The corrupted barcode s' is in the middle of end_f, $start_b$; then, the barcode satisfying $pro(s_0, s') = \max\{pro(s, s'), s \in \{barcode\}\}$ is used as the demultiplexing result.

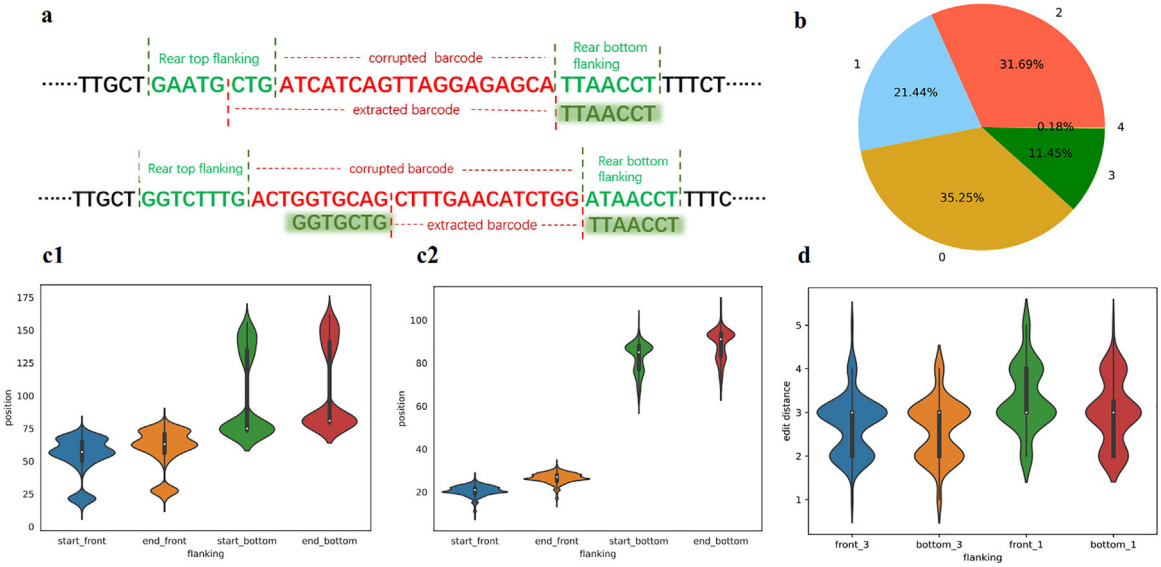


Fig. 3. Extracting barcodes accurately from sequencing sequences. (a) Extracting barcodes from $sequence'$, which can result in more errors in the extracted barcode. The barcodes ligated to the two sequences are 'ACTGGTGCAGCTTTGAACATCTGG' and 'ATGTCCCAGTTAGAGGAG GAAACA'. The rear top and bottom flanking sequences are 'GGTGCTG' and 'TTAACCT'. According to the flowchart in Fig. 2B, in the first sequence, $D_b = 0$. $D_f > 1$ and $D_b < D_f$, and the extracted barcode is $s' = sequence'[start_b - 24, start_b]$, which adds three insertion errors. For the other sequence, $D_b = D_f = 1$, the extracted barcode is $s' = sequence'[end_f + 1, start_b]$, which adds 10 deletion errors. The data used in Fig. (b), (c1), (c2) and (d) are from B3 (Table 2). (b) Pie chart with $D_m = \min\{D_f, D_b\}$; only 35.25% of the barcodes can be extracted accurately. Most barcode extractions are inaccurate and will add edit errors, as shown in Fig. 3a. (c1) Aligning the rear top flanking and the rear bottom flanking with $sequence'[0{:}120]$. A violin plot of the start and end positions of the two flanking sequences is shown for $D_b = D_f = 0$. (c2) Aligning the rear top flanking and rear bottom flanking with $sequence'[35 : !77]$ and $sequence'[35 + start_f : 120]$. A violin plot of the start and end positions of the two flanking sequences is shown for $D_b = D_f = 0$. (d) Violin plot of the edit distance of the flanking sequences and the barcode sequence in b3. Front 3 and bottom 3 represent the top and bottom flanking sequences of the barcoding library-kit PCR Barcoding Expansion. Front 1 and bottom 1 represent the top and bottom flanking sequences of the Native Barcoding Expansion after replacement. The flanking sequences in the Native Barcoding Expansion are more dissimilar to the barcodes in B3 than those in the PCR Barcoding Expansion.

3. Results

In this study, we developed a software package PRO that can be used to generate a barcode set and demultiplex the sequenced barcodes. In this section, we compare the barcode sets generated by PRO with randomly selected ones and compare the probability divergence with the edit distance in terms of demultiplexing accuracy. According to the tests, PRO exhibits superior performance in both generating barcode sets and demultiplexing. We ligated the barcodes generated by PRO and those provided by ONT onto the raw DNA sequence to generate simulated data and then demultiplexed them with both PRO and Guppy 6.5.7 to analyze the experimental results. PRO performed comparably to the official demultiplexing tool Guppy in terms of demultiplexing accuracy. In addition, we compared PRO with existing tools and validated its robustness using simulated data with varying sequencing accuracies.

3.1. Comparisons between the barcode set generated by PRO and the randomly selected sets

We used PRO to generate three barcode sets $\{barcode\}_{FPS}$ 15 bp, 20 bp, and 24 bp in length, each of which contained 106, 246 and 2,295 barcodes, respectively. As mentioned above, the core idea of the farthest-point sampling algorithm is to generate a barcode set such that the probability divergence between any two barcodes is as low as possible, which aims at more accurate demultiplexing. To demonstrate the power of the algorithm, we randomly selected as many sequences as were in $\{barcode\}_{FPS}$ from the preselected dataset, denoted by $\{barcode\}_{R}$. Then, we calculated the probability divergence between each pair of barcodes in $\{barcode\}_{FPS}$ and $\{barcode\}_{R}$. As shown in Fig. 4, it is clear that the mean probability divergence of $\{barcode\}_{FPS}$ was much lower than those of $\{barcode\}_{R}$ regardless of the barcode length and the size of the barcode sets. In addition, it contained fewer outliers, and the sequences corresponding to the outliers are generally error-prone in demultiplexing.

To validate PRO-dex in terms of demultiplexing accuracy, we simply simulate sequencing errors by randomly adding sequencing errors to barcodes in $\{barcode\}_{FPS}$ and $\{barcode\}_{R}$ and then employ PRO-dex to demultiplex the corrupted barcodes. In this research, we define the probability of divergence under the assumption that the insertion, deletion and substitution of each base occur independently during sequencing; thus, we simulated the procedure by adding edit errors to barcodes independently, as in previous research [8]. The error rate of ONT sequencing is approximately 90%, so we randomly added 6, 7 and 8 edit errors to the 24 bp barcodes; 5, 6 and 7 errors to the 20 bp barcodes; and 4, 5 and 6 errors to the 15 bp barcodes. Each barcode has errors added to it 100 times independently.

Here, we define the barcodes with added errors as corrupted barcodes; if a corrupted barcode can be assigned to the barcode that generated it, we say that the barcode is correctly demultiplexed. The demultiplexing accuracy is defined as the fraction of correctly demultiplexed barcodes. As shown in Fig. 5, the demultiplexing accuracy for $\{barcode\}_{FPS}$ is higher than that for $\{barcode\}_{R}$, regardless of the barcode length or sequencing errors. In addition, as the number of barcodes increases, the demultiplexing accuracy decreases, but the accuracy for $\{barcode\}_{R}$ decreases faster than that for $\{barcode\}_{FPS}$.

3.2. Comparisons between the probability divergence and the edit distance

To demonstrate the advantage of probability divergence, we replaced it with the edit distance and repeated the experiments in the last section. Fig. 5 demonstrates that the demultiplexing accuracy when using probability divergence is greater than that when using edit distance for both $\{barcode\}_{FPS}$ and $\{barcode\}_{R}$.

3.3. Performance evaluation of PRO

The greatest advantage of ONT sequencing is the long sequencing length; however, the sequencing error rate is relatively high. An

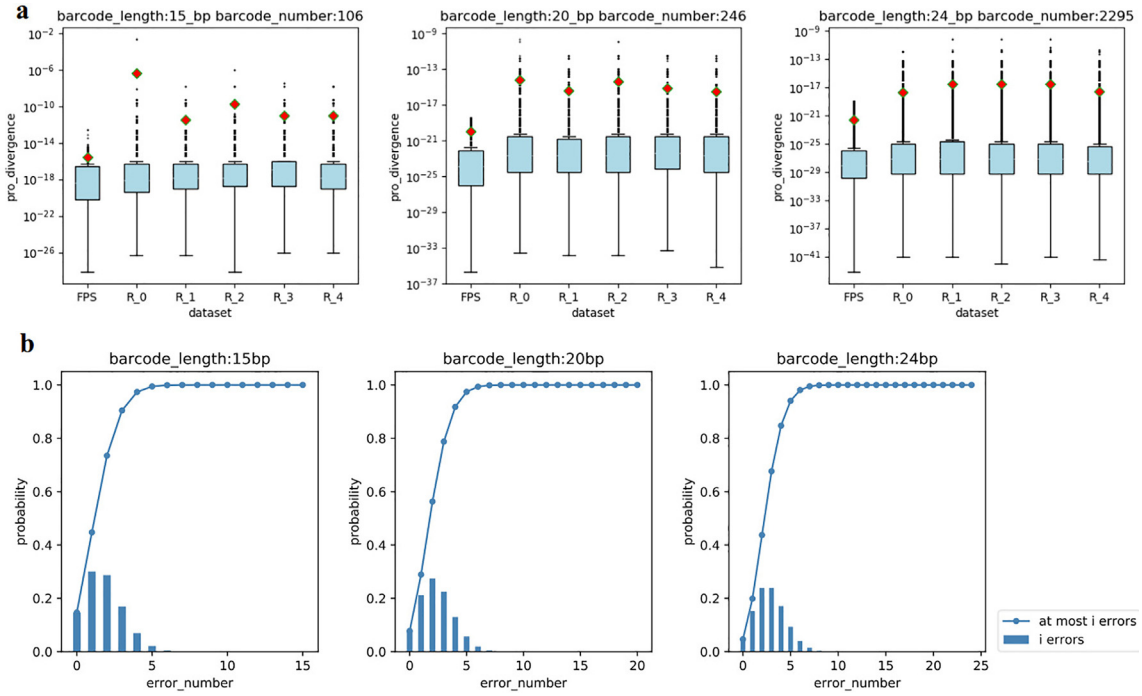


Fig. 4. For $\{barcode\}_{FPS}$ of different lengths, we randomly select five items from the prescreened dataset $\{dataset\}_{pre}$ to obtain five random barcode sets $\{barcode\}_{R}$ with the same size as $\{barcode\}_{FPS}$: $R0$, $R1$, $R2$, $R3$, and $R4$. The random seeds are 0, 1, 2, 3, and 4. (a) Box plots of the probability divergence between all pairs of barcodes in $\{barcode\}_{FPS}$ and $\{barcode\}_{R}$. The mean probability divergence of $\{barcode\}_{FPS}$ is smaller than that of $\{barcode\}_{R}$, and there are fewer outliers. (b) With a sequencing accuracy of ~90%, the bar chart shows the probability of i errors occurring in a DNA sequence during sequencing, and the solid line shows the probability of at most i errors occurring.

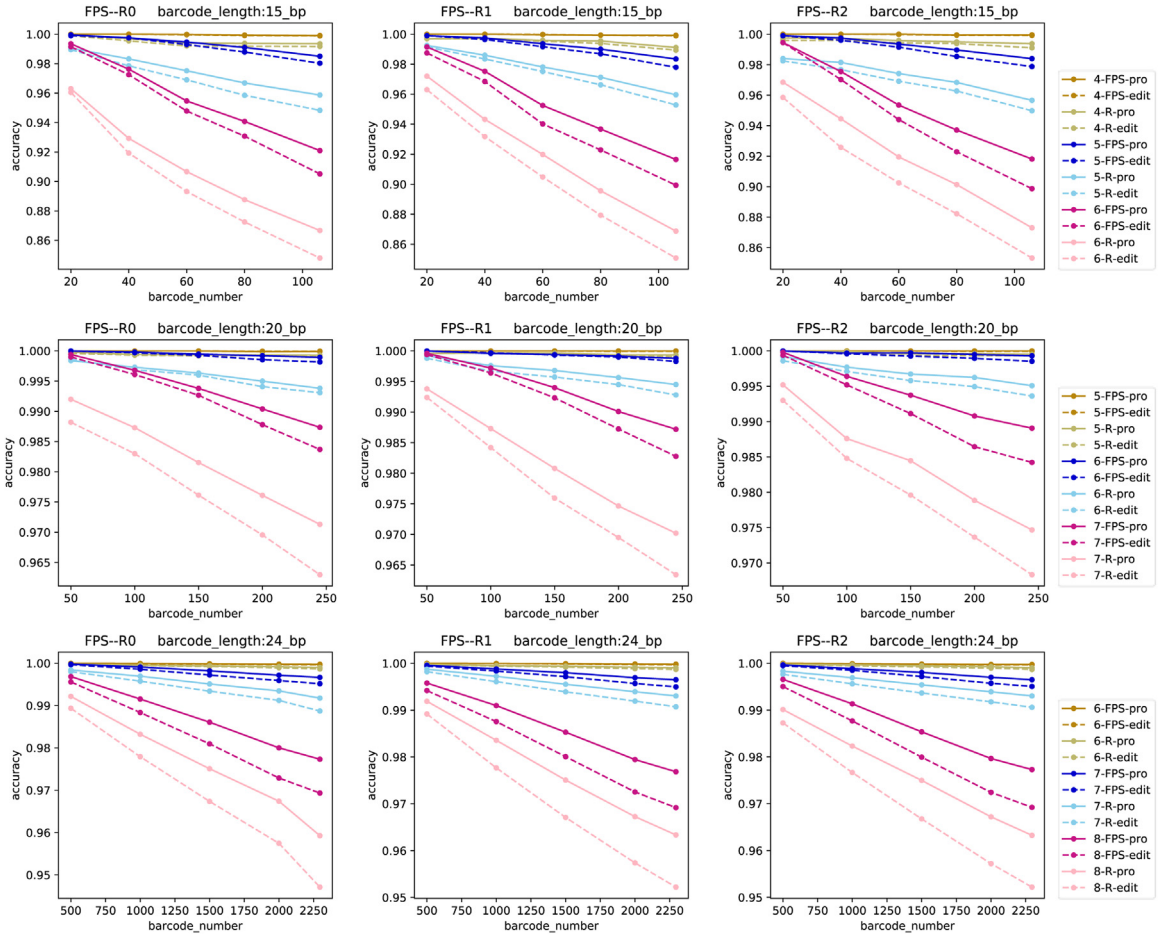


Fig. 5. Comparison of $\{barcode\}_{FPS}$ with $\{barcode\}_{R}$ and probability divergence with edit distance in terms of demultiplexing accuracy. $R0$, $R1$, and $R2$ are three random barcode sets, as described above. The line graphs of $R3$ and $R4$ are shown in supplementary material S2.1. The size of the 15 bp barcode set is 106, with 4, 5 and 6 edit errors added to each barcode sequence. The size of the 20 bp barcode set is 246, with 5, 6 and 7 edit errors added to each barcode sequence. The size of the 24 bp barcode set is 2,295, with 6, 7 and 8 edit errors added to each barcode sequence. The solid line shows the demultiplexing accuracy of the probability divergence, and the dashed line shows the demultiplexing accuracy of the edit distance. For the same barcode length and number of errors, $\{barcode\}_{FPS}$ achieves a higher demultiplexing accuracy than $\{barcode\}_{R}$, and probability divergence achieves a higher demultiplexing accuracy than edit distance.

ideal barcode set should contain as many barcode sequences as possible, and each pair of barcodes should be as distinguishable as possible, aiming at counteracting high error rates and improving identification. Therefore, tools to generate barcodes and demultiplex them are needed. However, to our knowledge, there seems to be no specific barcode generation strategy available for ONT sequencing. ONT offers barcode kits containing up to 96 barcodes, and the demultiplexing tool provided is Guppy, which is also the most widely used tool. Thus, we compared PRO with the barcode kit provided by ONT and the Guppy demultiplexing tool. In the simulation experiment, we simulated library preparation and sequencing for ONT via a multisample sequencing simulator (https://github.com/JustLeeee/ONT-sequencing-data-library-preparation-pipeline.git) and DeepSimulator 1.5 [22]. Then, we used PRO and Guppy 6.5.7 to demultiplex and evaluated the demultiplexing accuracy of both tools.

3.3.1. Test data generation

To generate the simulated data (Fig. 6), we first used a multisample sequencing simulator to prepare libraries for the original DNA samples to convert the samples into a format suitable for the nanopore sequencing device. The multisample sequencing simulator divided each DNA sample according to a mixed distribution to generate 100 reads, each of which was linked with the corresponding barcode in the barcode kit, the adaptor and the flanking sequences. After library preparation, DeepSimulator 1.5 was used to perform the simulation to obtain the final simulated sequencing data.

To extract the barcode sequence accurately when demultiplexing and avoid the scenario in Fig. 3a, we removed sequences in $\{barcode\}_{pre}$ that were similar to the flanking sequences, that is, sequences whose edit distances obtained by semiglobal alignment with flanking sequences were both less than or equal to 1. Then, PRO generated $\{barcode\}_{FPS}$ by the farthest-point sampling algorithm. We obtained five barcode sets $\{barcode\}_{FPS}$ using different random seeds, and the barcode set with 2,292 barcodes was selected in 24-mer space with seed 0. Table 2 shows the information for all the barcode kits used in the simulation experi-

Table 2
Detailed information on the barcode kit.

No.	$\{barcode\}$	size	Barcode length (bp)
B1	Native barcoding expansion (1–12)	12	24
B2	Barcoded 16 s kit	24	24
B3	PCR barcoding kit (96)	96	24
B4	ours	12	24
B5	ours	24	24
B6	ours	96	24
B7	ours	500	24
B8	ours	1,000	24
B9	ours	2,292	24

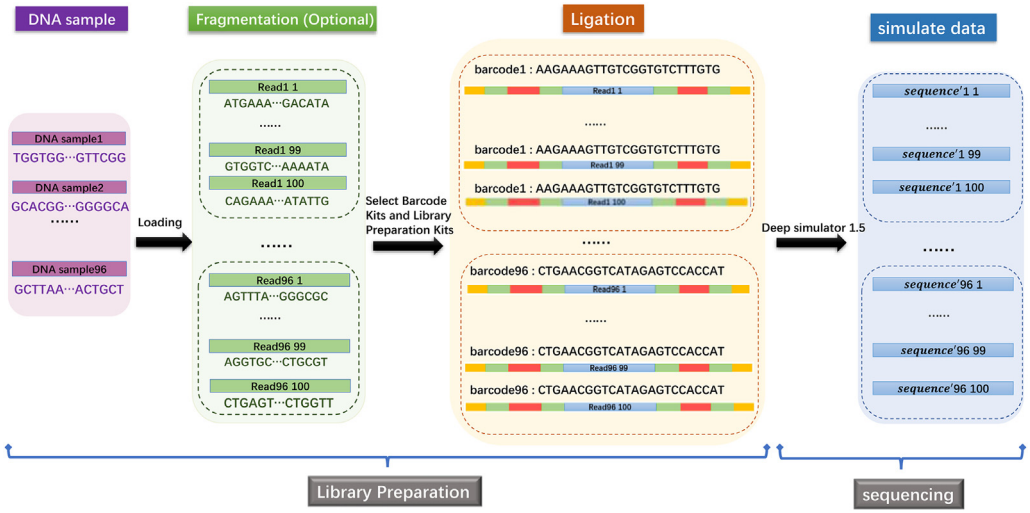


Fig. 6. Generation of nanopore sequencing simulation data. The DNA samples are first fragmented, and a mixture distribution is used to generate 100 reads per raw genome. The reading direction of the sequence is subsequently specified, and Barcode Kits and Library Preparation Kits are selected. The adapter sequences, flanking sequence and barcode are ligated to the read and input into DeepSimulator 1.5 to obtain the final sequencing simulation data.

ments. The results for the remaining four sets of $\{barcode\}_{FPS}$ are presented in Supplementary Material S2.2.

There were 105 raw DNA samples used to generate the simulation data, including 104 whole-genome shotgun sequences of Escherichia coli and 1 complete genome of Enterobacteria phage lambda. The details of the usage of PRO and DNA samples are presented in Supplementary Material S1 and Table S6.

3.3.2. Result analysis

We examined the barcodes generated by PRO in terms of both size and demultiplexing accuracy. The barcode kits used for the experiments consisted of B1–9. B1–3 were provided by ONT and contained the following barcode sequences: the Native Barcoding Expansion (1–12), the Barcoded 16 s Kit, and the PCR Barcoding Kit (96). B4–B9 were designed by PRO, of which B4–B8 are subsets of B9, meaning that B4 is made up of barcodes 1 to 12 of B9, and similarly for the others. After generating the simulated sequencing data, we demultiplexed the data using Guppy and PRO.

The size of the PRO barcode set can reach 2,292, which is much greater than the size of the official sets. In the 24-mer space, PRO designed a barcode set of size 2,292 (B9) with a demultiplexing accuracy of 98.29%, while the demultiplexing accuracy of Guppy on B9 was only 93.98%.

Guppy had excellent accuracy on the official barcode kits (B1–B3) provided by ONT, with a minimum accuracy of 99.66% (Table 3), while the accuracy of PRO was slightly lower, with an average decrease of 0.80% and a maximum difference of 1.66% on B3. The B4-B6 we designed were the same size as B1-B3 in the officially available barcode kits. On B4–B6, PRO showed superior demultiplexing accuracy, averaging 4.34% higher than that of Guppy and up to 4.73% higher on B4. Overall, the lowest demultiplexing accuracy for Guppy was 93.98% on B9, whereas the lowest demultiplexing accuracy for PRO was only 98.29% on B3. We determined that the reason for the poor performance of PRO on B3 was mainly due to the similarity of the flanking sequences to the barcodes in B3, which makes the flanking sequences likely to be aligned with the barcode sequences, ultimately resulting in inaccuracies in the extracted barcodes and low demultiplexing accuracy.

When demultiplexing after nanopore sequencing, the first step is to extract the barcode from $sequence'$. We analyzed the percentage of barcodes extracted accurately in B3, as shown in Fig. 3b, and this percentage was only 35.25%. Inaccurate extraction is not favorable for demultiplexing since it results in more errors in the extracted barcodes (Fig. 3a). To address this problem, we replaced the barcode library kit PCR Barcoding Expansion with Native Barcoding Expansion, a library kit whose flanking sequences are dissimilar (Fig. 3d) to the barcode sequences in B3, in the library preparation procedure, which aims to extract the barcodes accurately. In other words, we replaced the flanking sequences '*GGTGCTG TTAACCT*' with '*AAGGTTAA CAGCACCT*' without changing the barcodes in B3. After replacing the barcode library kit, PRO achieved a demultiplexing accuracy of 99.55%, while Guppy achieved a demultiplexing accuracy of only 96.26%.

In addition, the barcode kits (B7–B9) generated by PRO contained many more barcodes than did the official kits, and PRO exhibited higher demultiplexing accuracy on the generated barcodes than did Guppy. As the size of the barcode kits increased from B4 to B9, the demultiplexing accuracy of PRO gradually decreased, which is in line with theoretical expectations. The generation algorithm of PRO guarantees that the probability divergence between any two barcodes in a given barcode kit we design is low so that the demultiplexing accuracy of each barcode is high; that is, the demultiplexing process is stable. Fig. 7a shows the demultiplexing accuracy of PRO and Guppy for each barcode in B7–B9, from which we can see that the barcode kits (B7–B9) gener-

Table 3
Accuracy of different demultiplexing methods on barcode sets.

No.	Barcode kit size	Number of reads	Nanopore barcoding kit	Arrangement file	Accuracy of Guppy	Accuracy of PRO
B1	12	1,200	EXP-NBD104	barcode_arrs_nb12.cfg	100.00%	99.75%
B2	24	2,400	SQK-16S024	barcode_arrs_16 s.cfg	99.71%	99.21%
B3	96	9,600	EXP-PBC096	barcode_arrs_pcr96.cfg	99.66%	98.00%
B4	12	1,200	_	_	95.42%	100%
B5	24	2,400	_	_	96.21%	99.92%
B6	96	9,600	_	_	94.67%	99.40%
B7	500	50,000	_	_	93.89%	99.05%
B8	1,000	100,000	_	_	93.96%	98.74%
B9	2,292	229,200	_	_	93.98%	98.29%

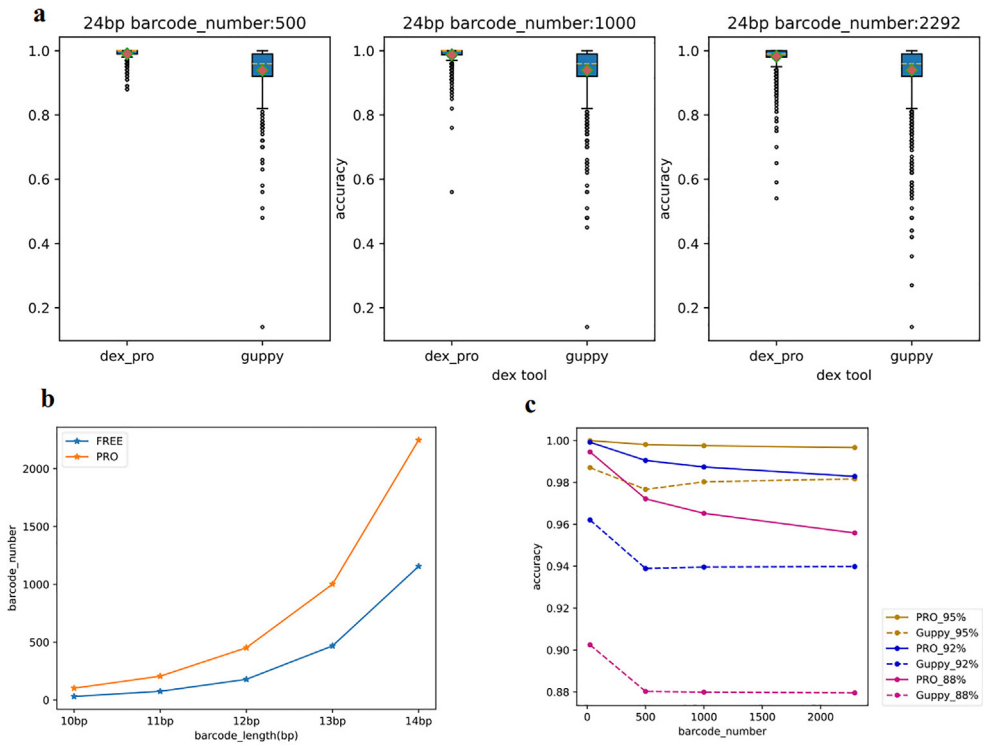


Fig. 7. Experimental results for validating the generation and demultiplexing effectiveness of PRO. (a) Box plot of single-barcode demultiplexing accuracy. (b) Numbers of barcodes of different lengths generated by PRO and FreeBarcodes. (c) Performance of PRO on simulated data with different levels of sequencing accuracy.

ated by PRO contain fewer outliers and that PRO is more stable than Guppy.

3.3.3. Additional tests

We subsequently compared PRO to a well-established and widely employed barcode design tool, namely, FreeBarcodes [8], which can generate(a large number of barcodes and be validated via next-generation sequencing data. First, we ligated the barcodes generated by PRO and FreeBarcodes onto the raw DNA sequences to generate simulated data. The simulated data were then demultiplexed with Guppy 6.5.7, based on which we evaluated the barcodes generated by PRO and FreeBarcodes.

FreeBarcodes consumed a large amount of computing resources, especially when generating longer barcodes. For instance, when generating barcodes of length 24 bp, similar to ONT's official kits, FreeBarcodes failed to generate results on our server (with approximately 1 TB of available memory) due to insufficient memory. Therefore, we evaluated the performance of PRO and FreeBarcodes in generating only barcodes with shorter lengths, which ranged from 10 bp to 14 bp. Furthermore, for the sake of fairness, we utilized Guppy 6.5.7 to conduct the demultiplexing process for the simulated data with the barcodes from both PRO and FreeBarcodes.

As shown in Fig. 7b, for the same barcode length, PRO generated more barcodes than FreeBarcodes. To evaluate the demultiplexing accuracy, we selected an equal number of barcodes to that generated by FreeBarcodes from the set generated by PRO and then generated simulated data and demultiplexed them using Guppy 6.5.7. Based on the results (Table 4), the PRO barcodes demonstrated greater demultiplexing accuracy than those from FreeBarcodes did.

In addition, to validate the robustness of the barcode sets generated by PRO, we generated simulated data using B9 with different levels of sequencing accuracy (i.e., 95%, 92%, and 88%). Subsequently, we demultiplexed the simulated data using PRO and Guppy 6.5.7. The results demonstrated that PRO achieved a greater demultiplexing accuracy than did Guppy at the same level of sequencing accuracy in the simulated data (Fig. 7c).

Table 4
Demultiplexing accuracy for the barcodes generated by PRO and FreeBarcodes.

Tools	The lengths and counts of the generated barcodes				
	10 bp	11 bp	12 bp	13 bp	14 bp
	31	75	179	468	1,156
FREE	90.80%	92.00%	89.92%	89.44%	88.62%
PRO	93.30%	92.16%	91.36%	90.25%	89.65%

4. Discussion

Sequencing multiple samples in a single run using barcodes can maximize the utilization of resources and reduce costs. However, the official barcode kits provided by Oxford Nanopore Technologies do not support simultaneous sequencing of more than 96 samples in a single run; therefore, some studies have to make some trade-offs [17]. The high error rate of nanopore sequencing makes the design and demultiplexing of barcodes difficult.

To address this difficulty, we theoretically analyzed the problem of barcode generation and demonstrated that finding the largest barcode set in a DNA sequence space of the same length is an NP-complete problem. Based on the principle that barcodes should be sufficiently different from each other, we use a heuristic algorithm, the farthest point sampling algorithm, to generate barcodes. We propose using probability divergence, which is more accurate than edit distance, and design a software package PRO based on this idea. PRO allows users to generate their own barcode sets and demultiplex sequencing files for nanopore sequencing. In the 24-mer space, PRO generated 2,292 barcodes with a demultiplexing accuracy of 98.29%, greatly expanding capacity and improving the demultiplexing accuracy. When the barcode kits we de-

signed had the same size as the official barcode kits, we could achieve demultiplexing accuracy comparable to that of the official kits. PRO slightly underperformed Guppy on B1, B2 and B3 provided by ONT for the following two reasons: (1) the flanking sequences are similar to the barcode, and PRO cannot accurately extract them, which is a vital point to consider when designing barcode kits. (2) The strategy of ONT for designing B1, B2, and B3 differs from ours and is unable to ensure that the probability of divergence between barcodes is large enough.

Calculating probabilistic divergence involves backtracking using dynamic programming and then calculating the probabilities. The farthest-point sampling algorithm is a greedy algorithm, which means that the generation and demultiplexing of PRO remain time consuming even if the algorithm is adapted.

The rapid improvements in long-read sequencing technologies present a significant challenge for barcode design methods. As the throughput of long-read sequencers continues to increase, larger barcode kits are essential. The theory for generating and demultiplexing barcodes proposed in this research can be employed for both ONT and PacBio sequencing data; however, no tests on PacBio data have been performed yet. We expect PRO to be helpful for achieving a higher throughput for long-read sequencing, and we anticipate that PRO will facilitate life science research, such as single-cell analysis and transcriptome analysis for direct RNA sequencing of many species.

Author Contributions

R.H. conceived the idea for the project. R.Z. performed the experiments and manuscript draft. T.Y., G.L. and X.G. edited and revised the manuscript. All authors gave final approval of the version to be published and have contributed to the manuscript.

Declaration of competing interest

The authors declare that they have no conflicts of interest in this work.

Acknowledgments

This work was supported by the National Key R&D Program of China (2020YFA0712400) and the National Natural Science Foundation of China (11931008, 12101368, and 61771009), and the Natural Science Foundation of Shandong Province with (ZR2021QA013). The funders had no role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Code access

The source code for the latest version of the PRO package is available at https://github.com/rztongr/PRO.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.fmre.2024.04.014.

References

[1] R. Zilionis, J. Nainys, A. Veres, et al., Single-cell barcoding and sequencing using droplet microfluidics, Nat. Protoc. 12 (2017) 44–73.
[2] C. Plesa, A.M. Sidore, N.B. Lubock, et al., Multiplexed gene synthesis in emulsions for exploring protein functional landscapes, Science 359 (2018) 343–347.
[3] G. Zimmermann, D. Neri, DNA-encoded chemical libraries: Foundations and applications in lead discovery, Drug Discov. Today 21 (2016) 1828–1834.
[4] M. Hamady, J.J. Walker, J.K. Harris, et al., Error-correcting barcoded primers for pyrosequencing hundreds of samples in multiplex, Nat. Methods 5 (2008) 235–237.
[5] E. Lyons, P. Sheridan, G. Tremmel, et al., Large-scale DNA barcode library generation for biomolecule identification in high-throughput screens, Sci. Rep. 7 (2017) 13899.
[6] T. Buschmann, L.V. Bystrykh, Levenshtein error-correcting barcodes for multiplexed DNA sequencing, BMC Bioinform. 14 (2013) 1–10.
[7] P.I. Costea, J. Lundeberg, P. Akan, TagGD: Fast and accurate software for DNA Tag generation and demultiplexing, PLoS ONE 8 (2013) e57521.
[8] J.A. Hawkins, Jr SK Jones, I.J. Finkelstein, et al., Indel-correcting DNA barcodes for high-throughput sequencing, Proc. Natl. Acad. Sci. 115 (2018) E6217–E6226.
[9] B. Wang, Q. Zhang, X. Wei, Tabu variable neighborhood search for designing DNA barcodes, IEEE Trans. Nanobiosci. 19 (2019) 127–131.
[10] V.I. Levenshtein, in: Binary Codes Capable of Correcting deletions, insertions, and Reversals, Soviet Union, 1966, pp. 707–710.
[11] X. Luo, X. Kang, A. Schönhuth, Phasebook: Haplotype-aware de novo assembly of diploid genomes from long reads, Genome Biol. 22 (2021) 1–26.
[12] M. Singh, G. Al-Eryani, S. Carswell, et al., High-throughput targeted long-read single cell sequencing reveals the clonal and transcriptional landscape of lymphocytes, Nat. Commun. 10 (2019) 3120.
[13] K. Lebrigand, V. Magnone, P. Barbry, et al., High throughput error corrected Nanopore single cell transcriptome sequencing, Nat. Commun. 11 (2020) 4025.
[14] K. Sahlin, P. Medvedev, Error correction enables use of Oxford Nanopore technology for reference-free transcriptome analysis, Nat. Commun. 12 (2021) 2.
[15] N.M. Davidson, Y. Chen, T. Sadras, et al., JAFFAL: Detecting fusion genes with long-read transcriptome sequencing, Genome Biol. 23 (2022) 1–20.
[16] X. Fan, D. Tang, Y. Liao, et al., Single-cell RNA-seq analysis of mouse preimplantation embryos by third-generation sequencing, PLoS Biol. 18 (2020) e3001017.
[17] I. Gupta, P.G. Collier, B. Haase, et al., Single-cell isoform RNA sequencing characterizes isoforms in thousands of cerebellar cells, Nat. Biotechnol. 36 (2018) 1197–1202.
[18] L. Tian, J.S. Jabbari, R. Thijssen, et al., Comprehensive characterization of single-cell full-length isoforms in human and mouse with long-read sequencing, Genome Biol. 22 (2021) 1–24.
[19] Y. You, Y.D.J. Prawer, D. Paoli-Iseppi, et al., Identification of cell barcodes from long-read single-cell RNA-seq with BLAZE, Genome Biol. 24 (2023) 1–23.
[20] J. Hartmanis, Computers and intractability: A guide to the theory of np-completeness (michael r. garey and david s. johnson), Siam Rev. 24 (1982) 90.
[21] M. Sosic, M. Sikic, Edlib: A C/C plus plus library for fast, exact sequence alignment using edit distance, Bioinformatics 33 (2017) 1394–1395.
[22] Y. Li, S. Wang, C. Bi, et al., DeepSimulator1. 5: A more powerful, quicker and lighter simulator for Nanopore sequencing, Bioinformatics 36 (2020) 2578–2580.

Ting Yu is an associate professor at Shandong University. His current research interests are transcriptome assembly, and long read sequencing data analysis.

Xin Gao is a professor of computer science, acting associate director of Computational Bioscience Research Center (CBRC), and Lead of the Structural and Functional Bioinformatics Group at King Abdullah University of Science and Technology (KAUST). His current research interests are bioinformatics, computational biology, machine learning, and big data.

Guojun Li (BRID: 09023.00.61510) is a professor at Shandong University. His current research interests are combinatorial optimization, genome assembly, and single cell clustering.

Renmin Han (BRID: 07372.00.36612) is a professor at Shandong University. His current research interests are medical imaging processing, high performance computing, and machine learning.