

# Greedy Heuristics for Degenerate Primer Selection<sup>\*</sup>

Sudha Balla, Sanguthevar Rajasekaran and Ion Mandoiu

Dept. of Computer Sci. & Eng., University of Connecticut, Storrs CT 06269, USA

**Abstract.** Amplification of multiple DNA sequences in a single Polymerase Chain Reaction (PCR) experiment, called Multiplex PCR (MP-PCR), is a widely known laboratory technique that requires efficient selection of a set of synthetic strings called Degenerate Primers to result in successful PCR products. The Multiple Degenerate Primer Selection Problem (MDPSP) is a fundamental problem in molecular biology and has received attention from numerous researchers in the recent past. Several variants of MDPSP have been proved to be NP-Complete. Many heuristic algorithms exist in the literature for MDPSP that have been shown to perform well in practice on real biological data. In this paper, we present two new greedy heuristics for the problem, analyze their time and space complexities and compare the performance of one of the proposed algorithms on random and real biological data with that of two previously reported algorithms. Our results show that the proposed algorithm performs almost equally in terms of quality, i.e., the number of degenerate primers, and executes in time much less than the two algorithms, also eliminating the dependency on an empirical input parameter that the previous algorithms exhibit in their performance both in terms of quality and time of execution.

**Keywords:** Degenerate Primers, Primers for MP-PCR, Multiplex Primers

## 1 Introduction

PCR is a fundamental laboratory method in molecular biology that amplifies a given DNA sequence into exponential number of copies. This method requires two synthetic strings, called *primers*, typically 15-30 nucleotides in length that are essentially substrings on either ends of the region desired to be amplified, the 5' end and the 3' end, aptly known as the *forward* and the *reverse* primers respectively. MP-PCR [3] is an advanced technique that amplifies several DNA sequences in a single experiment. This necessitates the presence of a forward and a reverse primer for each of the DNA sequence to be amplified in a single reaction. Undesired events such as mispriming and primer cross-hybridization occur due to the presence of numerous primers during MP-PCR, when the number of input DNA sequences is large, compromising the quality of the PCR products.

---

<sup>\*</sup> This research has been supported in part by the NSF Grant ITR-0326155.

Hence, it is critical to minimize the number of primers employed in a MP-PCR. One approach is to select primers that act as forward and/or reverse primers to several of the input DNA sequences. Another method involves the introduction of multiple nucleotides in several positions of the synthetic primers (called *degenerate primers* in the literature) [4].

The *degeneracy* of a degenerate primer  $p$  is the product of the number of nucleotides in each of its positions. If  $|p| = l$ , then, its degeneracy  $d = \prod_{i=1}^l |p_i|$ . Usually, a bound on the degeneracy of the primers to be used in a MP-PCR is imposed as primers of very high degeneracy could result in numerous mispriming events. Therefore, the focus now shifts to identifying a set of degenerate primers, each of a specified length  $l$  and of at most the bound degeneracy  $d$ , that collectively *cover* the input sequences. A degenerate primer  $p$  is said to *cover* an input sequence  $s$  if one of the  $d$  non-degenerate primers represented by  $p$  is a substring of  $s$ . Many algorithms proposed in the literature select such primers one at a time, attempting to maximize its *coverage*, i.e., the number of sequences that it covers from the set of uncovered sequences. Formally, the problem can be stated as follows:

**Multiple Degenerate Primer Selection Problem (MDPSP):** Given  $n$  DNA sequences  $S = \{S_1, S_2, \dots, S_n\}$ , each of average length  $m$ , and also two integers  $l, l \ll m$  and  $d$ , find a set of degenerate primers  $P_d$  of minimum cardinality that covers  $S$ , such that each degenerate primer  $p \in P_d$  is of length  $l$  and degeneracy at most  $d$ .

A special case of MDPSP called Minimum Primer Degenerate Primer Design (MP-DPD) was formulated and proved to be NP-Complete in [5], where for  $|S_i| = l, 1 \leq i \leq n$ .

Several heuristic algorithms for MDPSP are available in the literature. Section 2 discusses a few of the salient ones in some detail. In 3 we propose two new greedy heuristics to design degenerate primers. Section 4 provides the runtime and space requirements of the algorithms proposed. Section 5 provides a comparative discussion on the performance of one of the proposed algorithms with two previously known algorithms with respect to quality of output and time of execution. Section 6 concludes the paper.

## 2 Algorithms for Degenerate Primer Selection

The first efforts of reducing the number of primers for multiple DNA sequence amplification by identifying common substrings in a subset of the input came from Pearson et al. in [7], where the authors designed a set of non-degenerate primers using a greedy set cover algorithm and an exact branch-and-bound algorithm. Linhart and Shamir pioneered the work on degenerate primer design in [6], proposing algorithm HYDEN for MDPSP. HYDEN designs every degenerate primer by repeatedly solving the Maximum Coverage Degenerate Primer Design (MC-DPD) problem on the uncovered set of input sequences. The authors showed that HYDEN performed well in practice when used to design degenerate primers for a set of Human Olfactory genes that in turn were used to

extract genes of the same family from genome data. Their elaborate work in [5] formulates and proves several variants of Degenerate Primer Selection to be NP-Complete. Wei, Kuhn and Narasimhan [10] proposed algorithm DePiCt that employs hierarchical clustering to group a set of given protein sequences based on similarity and designs a forward and a reverse primer for each cluster from highly conserved regions on a multiple alignment of the sequences of the cluster.

Algorithm MIPS [9] follows an iterative beam search technique to design degenerate primers. It starts with a set of primers (called *2-primers*) that cover two sequences from an input of  $n$  sequences. Then it extends the coverage of the primers in the candidate set by one additional sequence, introducing degeneracy in the primers if necessary, retains a subset of these primers (the number determined by an input parameter called *beam size*  $b$ ) for the next iterative step until none of the primers can be extended further without crossing the target degeneracy. At this point, the primer with the lowest degeneracy is selected and the sequences that it covers (let the number be  $k$ ) are removed from the input set and the procedure is repeated until all the sequences are covered. MIPS has overall time complexity of  $O(bn^3mp)$ , where  $b$  is the beam size,  $n$  is the number of sequences,  $m$  is the sequence length,  $l$  is the primer length, and  $p$  is the cardinality of the final set of selected degenerate primers. Experimental results for varying number of input sequences and different target degeneracy, the sequences being uniformly distributed i.i.d. sequences of equal length, were reported in [9]. It was shown that MIPS always produced lesser number of primers than HYDEN.

Algorithm DPS [1] follows the footsteps of MIPS but uses additional sorting and a new ranking metric called coverage-efficiency in each iterative step to order and select the  $b$  best primers, thus improving the worst-case time complexity of MIPS. DPS was shown to perform better in practice on real biological data both in quality and time when compared to MIPS.

A downside of both MIPS and DPS is that their runtime depends upon the empirical input parameter  $b$ ; the authors of [9] suggest using a value of  $b$  close to the number of sequences in the input to achieve a good trade-off between solution quality and runtime. Although DPS was shown to achieve better quality of output for lesser values of  $b$ , its dependency on  $b$  still exists. In the following section we propose two new greedy approaches that eliminate the dependency on  $b$ , while improving the runtime and retaining the quality of output. The greedy approaches avoid the time consuming process of explicit generation of next generation candidates from substrings of length  $l$  from all the uncovered sequences as done by MIPS and DPS. Instead, in order to increase coverage, they identify the best candidate to merge the primer with, based on the *hamming distance* and *degenerate increase potential* of possible candidates of the input, explained in detail in the next section. Also, in order to speed up the generation of the initial set of 2-primers, MIPS and DPS adopt a FASTA look-up approach, which could skip some valid primer candidates because of the mismatches between two substrings of length  $l$  being distributed in such a way that they do not share a consecutive substring of length  $k$ -tup. This is overcome by the greedy algorithms by adopting an efficient technique to calculate

the hamming distance and degenerate increase potential between substrings of length  $l$  in a sequence and those of other sequences of the input.

Algorithms HYDEN, MIPS and DPS address MDPSP, while in [2] a single degenerate primer for each end of the sequences to be amplified is designed attempting to minimize the degeneracy of the primer designed. MinDPS of [2] addresses the Minimum Degeneracy Degenerate Primer Design (MD-DPD) problem described in [5], [6].

### 3 New Greedy Algorithms for MDPSP

#### 3.1 Algorithm DPS-HD

In this section we propose a new greedy heuristic for MDPSP, called algorithm DPS-HD (stands for Degenerate Primer Selector by Hamming Distance). Similar to its predecessors, DPS-HD designs members of the output set  $P_d$  one at a time, attempting to select a degenerate primer  $p$  of maximum coverage for the set of input sequences to be covered yet. In order to select  $p$ , DPS-HD works as follows. Let us consider that we are selecting the first degenerate primer of  $P_d$ . Therefore, all the input sequences are alive (yet to be covered). An arbitrary sequence  $S_i, 1 \leq i \leq n$  is selected; let it be  $S_k$ . For every substring of length  $l, S_{k,j}, 1 \leq j \leq (m-l+1)$ , its hamming distance (i.e., the number of mismatches) with all substrings of length  $l$ , for  $S_{i,j}, 1 \leq i \leq n, 1 \leq j \leq (m-l+1)$ , is calculated using the efficient technique described in [8]. For every  $u$  that is a substring of length  $l$  in  $S_k$  the following is performed to develop  $u$  from a non-degenerate primer that covers  $S_k$  to a degenerate primer of degeneracy at most  $d$  to cover as many uncovered sequences as possible. Let  $c[u]$  be a binary array of size  $n$ . The  $k$ -th bit is set to 1 to indicate that  $u$  covers  $S_k$ . Let  $D[0:l]$  be an array of linked lists, where  $D[h], 0 \leq h \leq l$  represents the list that contains elements of the form  $(seq_{num}, position)$ , where the hamming distance  $dist(u, S_{seq_{num}, position}) = h$ .  $D[0]$  contains those elements that are sequences already covered by  $u$ , therefore, its elements are examined one at a time, and the  $seq_{num}$ -th bit of  $c[u]$  is set to 1 to cover additional sequences of the input covered by  $u$  in constant time. Then, an arbitrary element from the non-empty list of  $D$  with the lowest value of  $h$  is chosen and the substring of length  $l$  represented by the chosen element, say  $v$ , is merged with  $u$ .  $c[u]$  is updated accordingly. Let  $dist(u, v) = h_{uv}, 1 \leq h_{uv} \leq l$ . The  $h_{uv}$  positions to which additional symbols are added to  $u$  and the corresponding symbol added are maintained in separate lists. For  $D[h], h_{uv} \leq h \leq l$ , the elements in the lists are processed, creating a next generation set of candidate sites, say  $D'$ , by recalculating the hamming distance of the substrings of length  $l$  ( $l$ -mers) in sequences that are alive in  $O(nmh_{uv})$  time. This procedure is continued until the degeneracy of  $u$  reaches the target degeneracy  $d$  or all input sequences are covered. The best  $u$  of  $S_k$ , i.e. the degenerate primer developed as above with the maximum coverage, is selected as  $p$  and added to  $P_d$ . The sequences covered by  $p$  are eliminated and the procedure is repeated several times until all input sequences are covered. The pseudo code of the algorithm is given below:

Let  $d(u)$  represent the degeneracy of a string of length  $l$ .

```

Algorithm DPS-HD {
   $P_d := null$ ;
   $R := \{1, 2, \dots, n\}$ ; //sequences alive
  while( $|R| > 0$ ) {
     $selectedPrimer := null$ ;
     $selectedCoverage := 0$ ;
    (1) Choose an arbitrary sequence  $k$  from  $|R|$ ;
    (2) Calculate the hamming distance of all  $l$ -mers in  $S[k]$ 
        with the  $l$ -mers of other sequences alive;
    (3) for each  $l$ -mer  $u \in S[k]$  do {
        expand  $u$  to cover additional sequences,
        by merging  $u$  with a  $v$  at a minimum
        hamming distance from it, and recalculating
        hamming distances of sites alive,
        until  $d[u] := d$  or  $c[u] := |R|$ ;

        if( $c[u] > selectedCoverage$ ) {
           $selectedPrimer := u$ ;
           $selectedCoverage := c[u]$ ;
        }
      }
    (4)  $P_d := P_d \cup selectedPrimer$ ;
    (5) Eliminate sequences covered by  $selectedPrimer$  from  $R$ ;
  }
  output  $P_d$ ;
}

```

### 3.2 Algorithm DPS-DIP

In this section we propose another new greedy heuristic for MDPSP, called algorithm DPS-DIP (stands for Degenerate Primer Selector by Degenerate Increase Potential). This approach differs from DPS-HD by the criterion used to rank the possible primer sites that a given primer could be merged to expand its degeneracy and coverage, namely, the *degeneracy increase potential*, defined as follows:

**Definition 1.** The *degeneracy increase potential* (DIP) of a primer site  $v$  (a substring of length  $l$  in a sequence to be covered yet) is the factor  $f$  of increase in degeneracy that would effect by merging  $v$  with a primer  $u$ . If  $u' = u \cup v$ , then  $d[u'] = f \times d[u]$ .

Adopting DIP as the ranking criterion instead of the simple hamming distance of DPS-HD is based on the intuition that two primer sites, say  $x$  and  $y$ , that are at the same hamming distance from a primer  $u$  could result in primers

of different resultant degeneracy when merged with  $u$ , depending on the positions of  $x$  and  $y$  that differ from  $u$ . If the number of symbols in a position  $i$  of  $u$ ,  $1 \leq i \leq l$  is 1, if  $u[i] \neq x[i]$ , the potential increase in degeneracy  $x[i]$  would have is 2, if  $|u[i]| = 2$ , this value would be 1.5 and if  $|u[i]| = 3$ , it would be 1.33. The DIP factor  $f$  of a primer site  $x$  is the product of the potential increase of each of its positions with respect to the primer  $u$ . Therefore, we believe that ranking available primer sites in the non-decreasing order of their DIP values and selecting the site with the least DIP value to merge with  $u$  in order to expand its degeneracy and coverage could yield better quality primers for a given input set.

We use the same strategy of DPS-HD to calculate the DIP values of primer sites. As it is obvious, the DIP values of primer sites are real numbers in the range  $[1, 2^l]$ . A DIP value of 1 indicates the primer already covers a particular primer site (and in turn the sequence of the input that it is a substring of). If we adopt sorting methods such as the merge sort to order the primer sites with respect to their DIP values, we need to spend  $O(nm \log nm)$  time in each iteration step of expanding a primer  $u$ . We know that each position of a primer site can have potential increase of one of the following values: 1, 1.33, 1.5 and 2. If we scale these values as 1, 2, 3 and 4 respectively, the range of DIP values would be  $[1, 4^l]$  and radix sorting can be used to sort the  $O(nm)$  DIP values in  $O(nml/w)$  time,  $w$  being the word length of the computer. This avoids the  $O(\log nm)$  factor in the runtime, although it increases the runtime of each iterative step by a factor of  $O(l/w)$  (much small compared to the former for large datasets).

## 4 Analysis of DPS-HD and DPS-DIP

### 4.1 Analysis of Algorithm DPS-HD

Step (2) of the algorithm above that calculates the hamming distance of all the  $O(m)$   $l$ -mers in the chosen sequence takes  $O(nm^2)$  time and  $O(nm^2)$  space, using the technique described in [8]. A non-degenerate primer can be expanded into a degenerate primer of degeneracy at most  $d$  in  $O(|\Sigma| \log_{|\Sigma|} d)$  iterations (as the number of symbols that can be added to achieve the degeneracy  $d$  is in the range  $[\lceil \log_2 d \rceil : (|\Sigma| - 1) \lceil \log_{|\Sigma|} d \rceil]$ ). Each expansion iteration recalculates the hamming distances of  $O(nm)$  candidate sites that are alive. Therefore, the expansion of one non-degenerate primer takes  $O(|\Sigma| \log_{|\Sigma|} dnm)$ . To select a primer  $O(m)$  primers are expanded in Step (3), therefore, selecting one primer takes  $O(|\Sigma| \log_{|\Sigma|} dnm^2)$  time. If  $|P_d| = p$ , then the runtime of algorithm DPS-HD is  $O(|\Sigma| \log_{|\Sigma|} dnm^2p)$  and its space requirement is  $O(nm^2)$ .

### 4.2 Analysis of Algorithm DPS-DIP

DPS-DIP differs from DPS-HD by way of calculating and ranking available primer sites with respect to their DIP values. Calculating the DIP values will take the same time as calculating the hamming distances, but the sorting step

takes additional time as explained above. Therefore, the runtime of DPS-DIP is  $O(|\Sigma| \log_{|\Sigma|} dnm^2(l/w)p)$  and its space requirement is  $O(nm^2(l/w))$ , as each DIP value in the range  $[1, 4^l]$  could be expressed as  $[(2l/w) + 1]$  computer words.

## 5 Experimental Results

We have implemented DPS-HD in Java and tested its performance on random and real biological data, essentially the same datasets described in [1]. MIPS algorithm implemented in C++ was obtained from its authors and DPS algorithm was implemented in Java. All the implementations were run on a PowerEdge 2600 Linux server with 4 GB of RAM and dual 2.8 GHz Intel Xeon CPUs - only one of which is used by the sequential algorithms. A comparison of the performance of MIPS, DPS and DPS-HD are provided in tables 1, 2 and 3 below.

Two real biological datasets were considered. The first dataset is a set of 95 DNA sequences on which the algorithm MIPS was tested in [9]. Each sequence in the dataset has an SNP in it and the goal of the MP-PCR is to amplify the regions of every sequence that would include the SNPs in the amplified products. Thus, the input to the algorithms was a dataset of 190 sequences, each input sequence having two representatives in the input set, the first being the subsequence from the start to one position before the SNP and the second being the complement of the subsequence from one position after the SNP to the end of the sequence. The second dataset is a set of 50 human olfactory genes, which we received from the authors of algorithm HYDEN. Each sequence in this dataset was approximately 1 Kbp long. Taking the first 300 nucleotides and the complement of the last 300 nucleotides of each gene generated the input set for the experiment. Thus, the input set consisted of 100 sequences each of length 300. The length of the primer searched was 20, the value of  $b = n$  were adopted for MIPS and DPS and the degeneracy thresholds considered were 4096, 16384, 65536 and 262144 respectively.

We generated 10 random datasets for each value of  $n$  in 20, 40, 60, 80, 100, 120, 140, 160, 180 and recorded the average over all the 10 runs for each option of  $n$  sequences. The length of the sequences in the datasets  $m$  was 300 each. The value of  $b = n$  was adopted for MIPS and DPS, the length of the primer searched for was 15 and the experiments were run for degeneracy of 10000 and 100000 respectively.

## 6 Conclusion

In this paper, we proposed two new greedy algorithms for the problem of selecting multiple degenerate primers for use in MP-PCR. The proposed algorithms eliminate the dependency of previously known algorithms on an empirical input parameter that affects the runtime and quality of output. It was shown that one of the algorithms, when implemented, executes faster than the previously known algorithms providing almost the same quality of output on random and

**Table 1.** Performance on random datasets:  $l = 15, d = 10000$ 

# of seq ( $n$ )	# of primers MIPS	time(sec) MIPS	# of primers DPS	time(sec) DPS	# of primers DPS-HD	time(sec) DPS-HD
20	4	1.7	4	1.7	4	1.51
40	6.3	9	6	11.9	6	5.41
60	9	26	8.3	39.1	9	11.95
80	11	54.1	10.8	93.2	11	20.82
100	13.1	100.3	12.3	179.6	13.2	32.67
120	15.1	180.5	14.1	316.0	15	45.44
140	17	218.4	16.1	499.3	17	63.02
160	19.2	313.8	17.8	761.1	19	80.40
180	21	422.8	19.5	1103.0	21	125.47

**Table 2.** Performance on random datasets:  $l = 15, d = 100000$ 

# of seq ( $n$ )	# of primers MIPS	time(sec) MIPS	# of primers DPS	time(sec) DPS	# of primers DPS-HD	time(sec) DPS-HD
20	3	1.7	2.9	2.5	2.8	1.47
40	4	8.0	4	18.3	4	5.15
60	5.6	22.3	5	57.4	5.6	11.26
80	7	47.6	6	131.1	7	19.13
100	8	88.8	7	251.7	8	28.13
120	9	139.6	8	431.9	9	40.99
140	10	217.1	8.9	679.8	10	53.93
160	10.9	326.6	9.8	1039.9	11	68.53
180	11.9	435.3	10.4	1530.8	12	85.10

**Table 3.** Performance on biological datasets:  $l = 20$ 

Dataset	Degeneracy ( $d$ )	# of primers MIPS	time(sec) MIPS	# of primers DPS	time(sec) DPS	# of primers DPS-HD	time(sec) DPS-HD
Olfactory	4096	13	151	9	95.7	10	13.75
	16384	10	178	8	118.6	8	13.19
	65536	9	220	6	167.3	6	14.81
	262144	7	235	5	240.6	5	14.449
SNP	4096	56	683	54	604.2	56	31.581
	16384	48	789	45	596.7	46	26.048
	65536	39	829	36	693.9	38	20.74
	262144	31	842	30	903.6	31	19.535

real biological data. We are currently collaborating with molecular biologists to test the practical performance of the designed primers in MP-PCR. We believe that the low memory requirement and fast execution of the new algorithm will be useful in processing very large input sets. We also plan to implement and test the performance algorithm DPS-DIP. As can be seen, our algorithms are highly amenable to parallelization and we plan to test if parallel implementations lead to better quality primers as the primers could be developed from several sequences as seeds.

## References

1. S. Balla, S. Rajasekaran, I. I. Mandoiu, “Efficient Algorithms for Degenerate Primer Search”, *International Journal of Foundations of Computer Science* to appear.
2. S. Balla, S. Rajasekaran, “An Efficient Algorithm for Minimum Degeneracy Primer Selection”, *IEEE Transactions on Nanobioscience (IEEE-TNB)* to appear.
3. J. S. Chamberlain, R. A. Gibbs, J. E. Rainer, P. N. Nguyen, C. T. Casey, “Deletion screening of the Duchenne muscular dystrophy locus via multiplex DNA amplification”, *Nucleic Acids Research* **16** (1988) 11141–11156.
4. S. Kwok, S. Y. Chang, J. J. Sninsky, A. Wang, “A guide to the design and use of mismatched and degenerate primers”, *PCR Methods and Applications* **3** (1994) S39–S47.
5. C. Linhart, R. Shamir, “The degenerate primer design problem Theory and Applications”, *Journal of Computational Biology* **12(4)** (2005) 431–456.
6. C. Linhart, R. Shamir, “The degenerate primer design problem”, *Bioinformatics* **18(1)** (2002) S172–S180.
7. W. R. Pearson, G. Robins, D. E. Wrege, T. Zhang, “On the primer selection problem in polymerase chain reaction experiments”, *Discrete Applied Mathematics* **71** (1996) 231–246.
8. P. Pevzner, S. Sze, “Combinatorial Approaches to Finding Subtle Signals in DNA Sequences”, *Proc. of Eighth International Conference on Intelligent Systems for Molecular Biology* (2000) 269–278.
9. R. Souvenir, J. Buhler, G. Stormo, W. Zhang, “Selecting Degenerate Multiplex PCR Primers”, *Proc. 3rd Intl. Workshop on Algorithms in Bioinformatics (WABI)* (2003) 512–526.
10. X. Wei, D. N. Kuhn, G. Narasimhan, “Degenerate Primer Design via Clustering”, *Proc. of the 2003 IEEE Bioinformatics Conference (CSB)* (2003) 75–83.