Optimization of cDNA-AFLP experiments using genomic sequence data

Teemu Kivioja, Mikko Arvas, Markku Saloheim, Merja Penttilä and Esko Ukkonen

ABSTRACT

Motivation: cDNA amplified fragment length polymorphism (cDNA-AFLP) is one of the few genome-wide level expression profiling methods capable of finding genes that have not yet been cloned or even predicted from sequence but have interesting expression patterns under the studied conditions. In cDNA-AFLP, a complex cDNA mixture is divided into small subsets using restriction enzymes and selective PCR. A large cDNA-AFLP experiment can require a substantial amount of resources, such as hundreds of PCR amplifications and gel electrophoresis runs, followed by manual cutting of a large number of bands from the gels. Our aim was to test whether this workload can be reduced by rational design of the experiment.

Results: We used the available genomic sequence information to optimize cDNA-AFLP experiments beforehand so that as many transcripts as possible could be profiled with a given amount of resources. Optimization of the selection of both restriction enzymes and selective primers for cDNA-AFLP experiments has not been performed previously. The in silico tests performed suggest that substantial amounts of resources can be saved by the optimization of cDNA-AFLP experiments.

Availability: A Perl implementation of the optimization method is available upon request from the authors.

Contact: Teemu.Kivioja@vtt.fi

1 INTRODUCTION

Hybridization-based methods for genome-wide expression analysis, such as microarrays require sequence information about the whole transcriptome of the target organism. If such information is not available, differential display methods (Vos et al., 1995; Bachem et al., 1996), such as cDNA amplified fragment length polymorphism (cDNA-AFLP) (Breyne et al., 2002, 2003) and its variants like HiCEP (Fukumura et al., 2003), are among the few available choices that can be used at the genome-wide level to identify previously unknown genes that have interesting expression patterns (Breyne and Zabeau, 2001). Another advantage of cDNA-AFLP is its high sensitivity, i.e. the ability to detect low-abundance transcripts as well (Fukumura et al., 2003), which can be difficult with the microarray technology (Evans et al., 2002).

Although sequence information is not absolutely necessary for cDNA-AFLP experimental design, it is extremely useful for designing the experiment in such a way that the expression of as many genes as possible can be measured with reasonable experimental effort. Even when the complete transcriptome of the target organism is not known, there is often lots of sequence information available that can be utilized in experimental design, such as sequences of genes cloned from the organism, EST collections or a genome of a closely related organism. Especially now when careful manual genome annotation has trouble keeping up with the fast increase in raw sequence data, there is still need for cDNA-AFLP that is made more effective by using the available sequence data.

In differential display methods, the cDNA mixture synthesized from an mRNA sample is divided into small subsets called pools and the cDNAs in each pool are PCR amplified and separated using gel (or capillary) electrophoresis. When the above procedure is carried out for samples collected from different conditions, differences in the intensities of corresponding gel bands reflect the relative differences in the expression levels of genes. In cDNA-AFLP, the division into subsets is a result of using restriction enzymes and selective PCR: cDNAs are digested with two different restriction enzymes, adapters are attached to the specific ends of the resulting fragments, and the fragments are amplified using primers extended with additional selective nucleotides. Thus, for each selective primer pair only the fragments whose ends match the primer extensions get amplified and these fragments form a pool. Finally, the fragments in each pool are separated by electrophoresis.

If the sequences of some transcripts are available from the organism, their location on the gels can be predicted and thus some bands can be readily identified and their expression patterns recorded. On the other hand, an interesting yet unidentified band can be sequenced and this way novel genes relevant for the biological process being studied can be found. Thus, cDNA-AFLP is a flexible tool that can be used even when genomic sequence information is not complete yet all the available sequence information can be utilized.

Unfortunately, only one pair of enzymes does not in practice produce a fragment for every cDNA molecule that could be amplified and detected by electrophoresis. The fragments generated from a particular cDNA can be too long or too short to be revealed by the electrophoresis set-up or, if either of the enzymes does not have a
For the reasons given above, the number of transcripts that can be profiled with a given number of restriction enzymes and selective PCR amplifications depends crucially on the choice of restriction enzymes and selective PCR primers. A computer simulation provides a simple and inexpensive possibility to explore different options beforehand. Our method is based on simulating a cDNA-AFLP experiment in silico for the known genome of a related organism or for a known part of the target genome. The underlying assumption is that the real target genome has roughly the same characteristics as the sequence data available. Thus, if we can find efficient enzymes and selective primers for the given data, they are also likely to work well on the real target. Another application of this computational approach is presented by Kivioja (2004).

At least two programs, GenEST (Qin et al., 2001) and AFLPinSilico (also a web service) (Rombauts et al., 2003), simulate cDNA-AFLP for one enzyme pair and a set of sequences given as input. A web service (Bikandi et al., 2004) provides similar AFLP simulations for a large number of bacterial genomes. In silico simulations have also been used to test the efficiency of potential enzyme pairs and their combinations (Breyne et al., 2003; Fukumura et al., 2003) before the actual experiment.

In the above methods, the coverage of cDNA-AFLP is estimated simply by counting for each enzyme pair the coverage of the fragments which are in the length range of electrophoresis. The total coverage of several enzyme pairs is estimated simply as the union of the coverages of the best individual pairs. The question of the number of selective primer nucleotides needed for reasonable band separation is treated as a separate issue even though Rombauts et al. (2003) mention the idea of choosing the most informative combinations of selective nucleotides.

Our main contribution is to treat the choosing of the enzyme pairs and selective primers together as one optimization problem. Our model explicitly takes into account the selective PCR and the electrophoresis so that in an in silico experiment only the bands that are sufficiently far apart from other bands in the electrophoresis give information. This allows us to consider all combinations of enzyme pairs and subsets of selective primers as possible experimental designs. Unfortunately, finding the most efficient experimental design (the one that enables profiling the largest set of transcripts) from the vast space of possible designs turns out to be an NP-hard optimization problem. We devise an optimal algorithm for the special case where only one restriction enzyme pair is used and a heuristic algorithm for the general case that carefully exploits the combinatorial structure of the problem. Tests with several datasets suggest that the amount of pools needed to profile a particular portion of transcripts could be reduced in many cases by 25–50% by optimized selection of enzyme pairs and selective primers.

2 SYSTEMS AND METHODS

The goal of the optimization is to find an experimental design that enables profiling as many different transcripts, i.e. RNA products of genes, as possible with the available resources. The most costly parts of the cDNA-AFLP procedure are the running of electrophoresis gels and the manual extraction and purification of fragments from them for sequencing. Also, using more than a few different restriction enzyme pairs in an experiment would be impractical. For example, a specific primer has to be designed for the DNA ends generated by each enzyme and use of each new restriction enzyme pair requires an additional RNA sample.

In our computational model the cost of the experiment is the number of pools. We define a pool p as a subset of transcripts determined by a triple \((v_1, v_2, v)\) where \(v_1\) and \(v_2\) are the restriction enzymes with specified restriction sites and \(v = N_1 N_2 \ldots N_t\) is a sequence of selective nucleotides \(N \in \Sigma\), where \(\Sigma = \{A, T, G, C\}\). The selective nucleotides can be attached to either

---

**Fig. 1.** cDNA-AFLP with 3′ end capture (Breyne and Zabeau, 2001; Fukumura et al., 2003). After the digestion with first restriction enzyme, the biotin (B) labeled 3′-fragments are captured with the aid of streptavidin-coated magnetic beads (M) followed by the digestion with the second restriction enzyme and the removal of the 3′-fragments. Adapters have to be attached to the specific ends (colored) of the fragments to facilitate selective PCR (the selective nucleotides are marked as \(N_1, N_2, N_3\) and \(N_4\)).

restriction site in the cDNA molecule, all fragments lack one or both of the two specific ends needed for selective PCR. In addition, several cDNAs having the same nucleotides next to the restriction sites can produce fragments with the same length in which case these fragments end up in the same gel band and consequently do not give any useful information.

The cDNA-AFLP protocol has recently been improved so that only one fragment at most is obtained from each transcript (Breyne and Zabeau, 2001; Breyne et al., 2003). After the digestion with the first enzyme, the 3′-fragments of the cDNAs are captured. Only these are then digested with the second enzyme. This leads to at most one fragment per cDNA which has the specific ends of both enzymes. Obtaining at most one fragment from each transcript reduces the redundancy of the fragment pools and by reducing the total number of fragments it also reduces the number of selected nucleotides needed for reasonable separation of bands on the gels. Thus, we will concentrate on this variant of cDNA-AFLP which we will call the 3′-variant (illustrated in Fig. 1).
or both primers as long as the positions are fixed, i.e. the sequence \( s \) is a concatenation of two sequences: the selective nucleotides attached to the left primer and the selective nucleotides attached to the right primer (Fig. 1).

Including the pool \( p \) in an experiment means doing the following laboratory work: the cDNA sample digested with the enzyme pair \( (e_1, e_2) \) is amplified using primers matching the specific ends generated by the enzyme pair \( (e_1, e_2) \) and extended with the selective nucleotides \( s \). Finally, the fragments are run in an electrophoresis gel. The pool \( p \) covers a transcript \( t \) or \( t \in p \) if \( p \) can be used to profile \( t \). The profiling is possible if three conditions are met: (1) the enzyme pair \( (e_1, e_2) \) produces a fragment from \( t \) that is in the length range of electrophoresis, (2) the fragment has the Watson–Crick pairs of nucleotides \( s \) in the selective positions and (3) \( p \) produces no other fragments of the same length so that the fragment of \( t \) can be identified from the gel.

The number of the pools approximates well the amount of resources needed in the experiment as long as one does not use too many different enzyme pairs. Thus, we want to maximize the total coverage of the experiment with a given number of pools and enzyme pairs. We get the following optimization problem (we denote the set of all sequences from the alphabet \( \Sigma \) by \( 2^\Sigma \)).

**Pool selection.** Let \( T \) be a collection of transcripts, \( E \) a set of restriction enzymes with specified restriction sites, \([d_{\text{min}}, d_{\text{max}}] \) an electrophoresis length range and \( P \) the set of all pools each determined by a triple from the set \( E \times E \times 2^\Sigma \). In addition, let \( k \) and \( n \) be positive integers. Select from the set of all pools \( P \) at most \( n \) pools that use at most \( k \) different enzyme pairs such that the pools cover as many transcripts as possible.

In practice, we also restrict the maximum number of selective nucleotides in a primer pair to some small value \( l \). Then each individual pool can be easily computed as a preprocessing step. However, finding a set of pools that maximize the total coverage is not as easy. In the Supplementary Material, we prove by reduction from max \( k \)-cover problem (Feige, 1998) the following inapproximability result for pool selection. We say that an algorithm approximates pool selection within a ratio of \( \alpha < 1 \) if the number of transcripts covered by the algorithm is at least \( \alpha \)-fraction of the number of transcripts covered by the optimal solution.

**Theorem 1.** For any \( \epsilon > 0 \), pool selection cannot be approximated in polynomial time within a ratio of \( (1 - 1/e + \epsilon) \) unless \( P = NP \).

In the formula, \( e = 2.71828 \ldots \) is the base of the natural logarithm. Note, however, that our formulation of the pool selection problem includes several input parameters, such as the number of different restriction enzymes available and the length range of the electrophoresis equipment, that in reality get only values from some limited range. Therefore, the inapproximability result merely shows that one is unlikely to find a polynomial time algorithm that solves the pool selection optimally without restricting the values of some of the parameters. In Section 3.1, we give an efficient algorithm for the special case where the enzyme pair is given and in Section 3.2 we use it to build a heuristic algorithm for the general pool selection problem.

### 3 ALGORITHMS

#### 3.1 An algorithm for selecting primers for an enzyme pair

In this section, we assume that the enzyme pair of the pools is given and develop an efficient algorithm for selecting the selective nucleotides of the primers so that the coverage of the pools is maximized. We will call a sequence of selective nucleotides \( s \) a selective pattern, and since the enzyme pair is fixed we say that a selective pattern \( s \) covers a transcript \( t \) if \( t \) is in the pool determined by \( s \) and the fixed enzyme pair. The algorithm can be used to optimize an experiment that uses only one enzyme pair but our main goal is to get a building block for our general algorithm for pool selection.

The 3′-prime variant of cDNA-AFLP allows efficient selection of selective patterns if we make the following restriction which we will call the prefix property: if a selective pattern \( s \) is chosen, no prefix of \( s \) can be chosen. By a prefix we mean a sequence consisting of the first \( i \) contiguous nucleotides of \( s \) for some \( i < |s| \), for example, if we choose pattern ATG we cannot choose its prefix AT. In the experiment, the prefix property means that each fragment produced by a certain enzyme pair is amplified by at most one pair of selective primers because the property guarantees that every chosen selective pattern has at least one different nucleotide compared with other chosen patterns. In contrast, the fragments amplified by a selective pattern are a subset of the patterns amplified by its prefix. Thus, choosing both a pattern and its prefix would usually cause redundant work.

The prefix property helps to design an optimization algorithm based on dynamic programming because in this case the pools of an enzyme pair are disjoint. The digestion with an enzyme pair produces at most one fragment from each transcript and the selective pattern unambiguously determines the pool of that fragment. Therefore, if two selective patterns (neither is a prefix of another) cover transcripts so that the first covers \( c_1 \) transcripts and the second one \( c_2 \) transcripts, together they cover exactly \( c_1 + c_2 \) transcripts because no transcript can be covered by both.

Generally, let \( c_m \) be the number of transcripts covered by the selective pattern \( s \). We assume that the coverage of each individual selective pattern has been computed as a preprocessing step. Let \( c_m \) be the maximum number of transcripts covered by \( m \) selective patterns with the common prefix \( s \). Similarly, let \( c_m[|j] \) denote the maximum number of transcripts that can be covered by \( m \) such patterns that each of them has a prefix in the set of prefixes \( \Sigma \). In addition, let \( sN \) denote the pattern which is a concatenation of \( s \) and a nucleotide \( N \). If we allow at most \( l \) selective nucleotides, we can compute \( c_m[|j] \) from the recurrence

\[
c_m[|j] = \begin{cases} 0 & \text{if } m = 0, \\ c_m & \text{if } m = 1 \text{ and } |s| = l, \\ \max\{c_{m-1}[|j|], c_{m-1}[|j|], c_{m-1}[|j|], c_{m-1}[|j|]\} & \text{if } m > 1 \text{ and } |s| < l, \\ \max\{c_{m-1}[|j|], c_{m-1}[|j|] + c_{m-1}[|j|]\} & \text{if } m > 1 \text{ and } |s| < l, \\ \end{cases}
\]

where \( c_{m|j} \) can be computed in the same manner

\[
c_{m|j} = \max_{0 \leq j \leq m} c_{m|j} + c_{m|j} \]

and, finally, \( c_{m|j} \) can be computed directly from the coverages of longer patterns

\[
c_{m|j} = \max_{0 \leq j \leq m} c_{m|j} + c_{m|j}.
\]

The choice of the order in which the nucleotides are processed is arbitrary. The correctness of the recurrences (1–3) follows from the fact that because of the prefix property, the only way to choose more than one pattern with a common prefix \( s \) is to only take patterns longer than \( s \) and they differ by at least one nucleotide. Thus, the corresponding subsets are disjoint and the size of their union is the sum of sizes of individual subsets.

Our goal is to compute the maximum coverage possible using at most \( n \) selective patterns \( c_{m[|j]} = \max_{0 \leq j \leq m} c_{m[|j]} \) where \( e \) denotes a sequence of length zero. We cannot simply take the value of \( c_{m[|j]} \)
since the coverage is not guaranteed to be an increasing function of the number of patterns \(m\) because of the prefix property (e.g. pattern AT can cover more than AT and ATG together but because of our restriction we cannot choose the coverage of AT and ATG as value of \(c_{i,j}\)). A set of at most \(n\) patterns giving the optimal coverage can be collected afterwards using traceback if we store the index values giving the maximum coverages while evaluating the recurrences.

The formulae (1–3) can be evaluated in time \(O(n^2)\) for a selective pattern \(s\) if the values \(c_{i,n}[m]\) for all \(N \in \Sigma\) have already been computed. There are \(O(4^l)\) patterns of length \(\leq l\). Thus, the result \(c[n]\) can be computed in time \(O(4^l n^2)\) which is reasonable because \(l\) is in practice quite small, e.g. 4. In addition, the preprocessing step requires following time. Let \(M\) be the total length of all the transcripts in \(T\). The fragments produced by an enzyme pair can be computed in time \(O(M)\). Since there are at most one fragment per transcript that belongs to at most \(l\) pools determined by at most \(l\) nucleotides, all the fragments can be divided into \(l\) pools in time \(O(l |T|)\).

The dynamic programming approach generalizes to the case in which a pair of enzymes is used in both orders. Namely, the sample is first digested using the enzymes in the order \((e_1, e_2)\) and selectively amplified and then separately the same is done in the order \((e_2, e_1)\). Since the \(3'\)-fragment can only be PCR-amplified if the second enzyme has a restriction site closer to the \(3'\) end of the sequence than the first enzyme, both the two enzyme orders can never produce a fragment from a transcript. Therefore, two pools, one produced by \((e_1, e_2)\), another by \((e_2, e_1)\), are always distinct, i.e. they never contain the same transcripts.

To extend the dynamic programming algorithm, let \(c^1[m]\) be the already computed maximum coverage of at most \(m\) pools, where \(m = 1, \ldots, n\), for the ordered pair \((e_1, e_2)\) and \(c^2[m]\) be the corresponding coverage for the pair \((e_2, e_1)\). Then, the maximum number of transcripts covered by the usage of both ordered pairs denoted by \(c^{1,2}[n]\) can be obtained from the recurrence

\[
c^{1,2}[n] = \max_{\delta \in \{+,-\}} \left( c^1[m] + c^2[n - m] \right).
\]

The following theorem summarizes the results for selecting primers for a fixed enzyme pair.

**Theorem 2.** The optimal set of at most \(n\) selective patterns of length \(\leq l\) for an enzyme pair can be chosen in time \(O(M + l |T| + 4^l n^2)\) if no chosen selective pattern is allowed to be a prefix of another chosen pattern.

### 3.2 An algorithm for pool selection

Our heuristic algorithm is a modification of the greedy algorithm for max \(k\)-cover (Feige, 1998). As in the greedy max \(k\)-cover algorithm, we iteratively select the enzyme pairs that cover the largest number of yet uncovered transcripts until we have used all the \(k\) different enzyme pairs. However, compared with max \(k\)-cover we have the additional freedom to choose the number of \(n\) pools to be allocated for each of the \(k\) different enzyme pairs. Thus, when adding the \(i\)-th enzyme pair, we also consider different possibilities to divide the pools between the \(i-1\) pairs already chosen and the new one. We can do it efficiently if we have stored the largest set of transcripts we have been able to cover using \(i-1\) pairs and \(j\) pools for each \(j = 1, \ldots, n\). Then, the dynamic programming algorithm given in Section 3.1 can be used to compute the optimal coverage of the new enzyme pair in the set of yet uncovered transcripts simultaneously for all numbers of pools \(m\) where \(1 \leq m \leq n - j\). We have to modify the algorithm slightly: instead of directly taking \(c_i\), the number of transcripts covered by the selective pattern \(s\), we only count transcripts that have not been covered by the \(j\) pools already chosen.

The pseudo code of the greedy pool selection algorithm is given in Figure 2. We assume that the pools have been computed as a preprocessing step. The algorithm returns the largest subset of transcripts found, \(C_{\text{max}}\), that can be covered using at most \(k\) enzyme pairs and \(n\) pools. It stores the intermediate results in a table \(\text{MAXCOVER}\) whose element \(\text{MAXCOVER}[i, j, e]\) contains the largest set of transcripts found so far that can be covered with \(j\) different enzyme pairs and \(j\) pools. The procedure \(\text{DYNAMIC}\) implements the modified dynamic programming algorithm and its traceback: \(\text{DYNAMIC}(A, m, e_i, e_j)\) returns a table with \(m\) elements such that the \(j\)-th element contains the largest subset that can be covered from the set of transcripts \(A\) using \(j\) pools with the enzyme pair \((e_i, e_j)\). Again, the at most \(n\) pools covering the set of transcripts \(C_{\text{max}}\) can be collected using traceback if some additional bookkeeping is performed during the greedy algorithm. It can be shown that the total running time of the greedy algorithm is \(O(k|E|^2n^2(4^l n + |T|))\) assuming that \(n > l\).

### 4 IMPLEMENTATION

We have implemented the simulation model described in Section 2 and the greedy algorithm for pool selection described in Section 3 and tested them with yeast (Saccharomyces cerevisiae), plant (Arabidopsis thaliana) and human (Homo sapiens) sequence data sets. In both of the large-scale cDNA-AFLP studies reported recently only a single enzyme pair has been used but in both orders. Breyne et al. (2002, 2003) digested a tobacco sample (Nicotiana tabacum) with MseI and BstYI enzymes and first divided it into two parts by preamplifying with a BstYI primer with either T or C nucleotide at the 3’ end of the BstYI restriction site PuGATCPy. They then made selective amplification with all combinations of two and three
The reported coverage of cDNA-AFLP (they call their variant of the technique HiCEP) in the yeast _Saccharomyces cerevisiae_ by using the enzyme pair (MspI, MseI) with all combinations of four selective nucleotides. Fukumura _et al_ (2003) tested the coverage of cDNA-AFLP (they call their variant of the technique HiCEP) in the yeast _Saccharomyces cerevisiae_ by using the enzyme pair (MspI, MseI) with all combinations of four selective nucleotides.

We simulated experiments in which enzyme pairs were always used in both orders and the number of pools was between 128 and 512 as in the reported studies but we allowed more than one enzyme pair to be used. We compared the performance of our greedy algorithm with a simple strategy that we call fixed. As explained in Section 3, the greedy algorithm tries to select such a combination of enzyme pairs and selective nucleotides that as many transcripts as possible are covered. Thus, in a greedy design each restriction enzyme pair may be followed by a different number of PCR amplifications and selective primers may have different numbers of selective nucleotides. In contrast, the fixed design always contains all selective primers for the given number of selective nucleotides, as in the reported studies. For the fixed design, we chose those enzyme pairs that individually cover the largest number of transcripts with the given number of selective nucleotides.

We considered all pairs of 10 restriction enzymes with different restriction sites of length four (four-cutters). The enzymes used are listed in Table 1. We excluded enzymes that have ambiguous recognition sites or produce blunt-ended fragments. We concentrated on four-cutters because they gave better coverages than other enzymes in preliminary simulations (data not shown). When simulating electrophoresis we used parameters similar to ones used by Fukumura _et al_ (2003). We accepted a fragment if its length was between 40 and 700 bp and assumed that two fragments can be separated if their length difference was >0.5% of their total length. The maximum number of selective nucleotides in a primer was four.

The test results are summarized in Table 2. The reported coverage of a greedy design is the percentage of transcripts it covers from the dataset that was used to optimize the design. The results with the human dataset are shown in more detail in Figure 3. The greedy designs with 4 enzyme pairs and 512 pools are shown in Table 3. Interestingly, when using 512 pools the coverage was higher in the human dataset than in the much smaller yeast dataset, probably owing to the higher average length of the transcripts in the human dataset (2692 versus 1412 bp). The coverages in yeast were lower than that reported by Fukumura _et al_ (2003), possibly because our dataset did not include the 3′-untranslated regions (3′-UTRs).

### Table 1. The restriction enzymes used in the simulations and their restriction sites

<table>
<thead>
<tr>
<th>Restriction enzymes</th>
<th>Restriction sites</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaeI</td>
<td>CTAG</td>
</tr>
<tr>
<td>TruI1</td>
<td>TTAA</td>
</tr>
<tr>
<td>DpnII</td>
<td>GATC</td>
</tr>
<tr>
<td>MspI</td>
<td>CCGG</td>
</tr>
<tr>
<td>NlaIII</td>
<td>CATG</td>
</tr>
<tr>
<td>Tsp509</td>
<td>AATT</td>
</tr>
<tr>
<td>HinP1II</td>
<td>GCGC</td>
</tr>
<tr>
<td>Csp6I</td>
<td>GTAC</td>
</tr>
<tr>
<td>TaqI</td>
<td>TCGA</td>
</tr>
<tr>
<td>MaeII</td>
<td>ACGT</td>
</tr>
</tbody>
</table>

Additional selective nucleotides. Fukumura _et al_ (2003) tested the coverage of cDNA-AFLP (they call their variant of the technique HiCEP) in the yeast _Saccharomyces cerevisiae_ by using the enzyme pair (MspI, MseI) with all combinations of four selective nucleotides. We simulated experiments in which enzyme pairs were always used in both orders and the number of pools was between 128 and 512 as in the reported studies but we allowed more than one enzyme pair to be used. We compared the performance of our greedy algorithm with a simple strategy that we call fixed. As explained in Section 3, the greedy algorithm tries to select such a combination of enzyme pairs and selective nucleotides that as many transcripts as possible are covered. Thus, in a greedy design each restriction enzyme pair may be followed by a different number of PCR amplifications and selective primers may have different numbers of selective nucleotides. In contrast, the fixed design always contains all selective primers for the given number of selective nucleotides, as in the reported studies. For the fixed design, we chose those enzyme pairs that individually cover the largest number of transcripts with the given number of selective nucleotides.

We considered all pairs of 10 restriction enzymes with different restriction sites of length four (four-cutters). The enzymes used are listed in Table 1. We excluded enzymes that have ambiguous recognition sites or produce blunt-ended fragments. We concentrated on four-cutters because they gave better coverages than other enzymes in preliminary simulations (data not shown). When simulating electrophoresis we used parameters similar to ones used by Fukumura _et al_ (2003). We accepted a fragment if its length was between 40 and 700 bp and assumed that two fragments can be separated if their length difference was >0.5% of their total length. The maximum number of selective nucleotides in a primer was four.

The test results are summarized in Table 2. The reported coverage of a greedy design is the percentage of transcripts it covers from the dataset that was used to optimize the design. The results with the human dataset are shown in more detail in Figure 3. The greedy designs with 4 enzyme pairs and 512 pools are shown in Table 3. Interestingly, when using 512 pools the coverage was higher in the human dataset than in the much smaller yeast dataset, probably owing to the higher average length of the transcripts in the human dataset (2692 versus 1412 bp). The coverages in yeast were lower than that reported by Fukumura _et al_ (2003), possibly because our dataset did not include the 3′-untranslated regions (3′-UTRs).

In reality, we cannot expect the target sequences to be available for the optimization of the experiment design. Therefore, we also tested the in silico coverage of designs that were optimized using sequences from another organism. We used two closely related, recently sequenced filamentous fungi _Neurospora crassa_ (Galagan _et al_., 2003) (10,082 transcripts) and _Aspergillus nidulans_ (Aspergillus nidulans, 2003, http://www.broad.mit.edu) (9541 transcripts) as the target organisms. We used the greedy algorithm and sequence data from different organisms to choose a design with 3 enzymes and 256 pools (yeast and human datasets were as described above). Table 4 shows the percentage of _N. crassa_ and _A. nidulans_ transcripts that were covered by the designs.

The longest simulation took ~2 days with a 3 GHz PC with 2 GB of main memory running Linux. The implementation is currently written using Perl and BioPerl (Stajich _et al_., 2002). The running time could be significantly reduced by writing the most intensive parts in C. However, the computational costs are negligible compared with the laboratory resources required by the experiments.

### 5 DISCUSSION

In this study, we present methods to optimize the selection of pools, i.e. the unique sets of cDNA-AFLP fragments given by a certain restriction enzyme pair and certain selective nucleotides. The test results shown in Section 4 are encouraging even though with a particular number of enzyme pairs and pools the improvements in in silico coverage achieved by the optimized designs are fairly modest. Considering a large space of experimental designs is still worthwhile. Using only the pools giving good gene coverage (greedy design) allows covering the same amount of genes with a smaller work load than simply using all pools with a certain restriction enzyme combination and number of selective nucleotides (fixed design). For example, consider the fixed designs with 512 pools and 4 enzyme pairs (Table 2). The greedy algorithm is able to cover the same amount of transcripts using 4 enzyme pairs but considerably fewer pools in all sequence datasets studied: yeast 267 pools (48%
Table 2. *In silico* coverages of different experiment designs

<table>
<thead>
<tr>
<th>Organism</th>
<th>Transcripts</th>
<th>Pools(^a)</th>
<th>Transcripts covered by the design (%)</th>
<th>Greedy(^b)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No. of enzyme pairs</td>
<td></td>
<td>No. of enzyme pairs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 2 3 4</td>
<td>1 2 3 4</td>
<td></td>
</tr>
<tr>
<td><em>S.cerevisiae</em></td>
<td>6355(^d)</td>
<td>128 (112)</td>
<td>51</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>256 (217)</td>
<td>—</td>
<td>72</td>
</tr>
<tr>
<td></td>
<td></td>
<td>384 (297)</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>512 (346, 267)</td>
<td>57</td>
<td>—</td>
</tr>
<tr>
<td><em>A.thaliana</em></td>
<td>18590(^e)</td>
<td>128 (116)</td>
<td>47</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>256 (226)</td>
<td>—</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td></td>
<td>384 (313)</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>512 (512, 374)</td>
<td>62</td>
<td>—</td>
</tr>
<tr>
<td><em>H.sapiens</em></td>
<td>16138(^f)</td>
<td>128 (116)</td>
<td>51</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>256 (209)</td>
<td>—</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td></td>
<td>384 (268)</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>512 (433, 329)</td>
<td>65</td>
<td>—</td>
</tr>
</tbody>
</table>

\(^a\)The number of pools used. The number of pools in the greedy design that covers as many genes as the fixed design is given in the parentheses (with 512 pools the first figure corresponds to 1 enzyme pair and the second one to 4 enzyme pairs).  
\(^b\)In *silico* coverage (%) using all primers with a fixed number of selective nucleotides (3 when 128..384 pools, 3 or 4 when 512 pools), enzyme pairs used in both orders.  
\(^c\)In *silico* coverage (%) when using the pools chosen by the greedy algorithm, enzyme pairs used in both orders.  
\(^d\)All ORF coding sequences from SGD version dated June 23, 2003, no UTRs.  
\(^e\)All transcripts from UniGene build 42 that have a RefSeq identifier.  
\(^f\)All transcripts from UniGene build 166 that have a RefSeq identifier.

Table 3. The greedy designs using 4 enzyme pairs and 512 pools

<table>
<thead>
<tr>
<th>Organism</th>
<th>Enzyme pair</th>
<th>Number of primers</th>
<th>1st Order</th>
<th>2nd Order</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>S.cerevisiae</em></td>
<td>(DpnII GATC, Csp6I GTAC)</td>
<td>64</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(TaqI TCGA, NlaIII CATG)</td>
<td>62</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Tru1I TTAA, MaeII ACGT)</td>
<td>27</td>
<td>91</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Tru1I TTAA, Tsp509 AATT)</td>
<td>68</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td><em>A.thaliana</em></td>
<td>(Tru1I TTAA, Mspl CCGG)</td>
<td>18</td>
<td>115</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(MaeI CTAG, TaqI TCGA)</td>
<td>64</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(MaeII ACGT, NlaIII CATG)</td>
<td>73</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(TaqI TCGA, Csp6I GTAC)</td>
<td>71</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td><em>H.sapiens</em></td>
<td>(MaeI CTAG, DpnII GATC)</td>
<td>45</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Tru1I TTAA, Mspl CCGG)</td>
<td>16</td>
<td>108</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(DpnII GATC, Csp6I GTAC)</td>
<td>69</td>
<td>85</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(MaeI ACGT, NlaIII CATG)</td>
<td>103</td>
<td>37</td>
<td></td>
</tr>
</tbody>
</table>

Table 4. The *in silico* coverages of the designs optimized with another organism (3 enzyme pairs and 256 pools)

<table>
<thead>
<tr>
<th>Target organism</th>
<th>Design organism</th>
<th>Coverage (%)(^a)</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>N.crassa</em>(^b)</td>
<td><em>N.crassa</em></td>
<td>76</td>
</tr>
<tr>
<td><em>A nidulans</em>(^c)</td>
<td><em>A nidulans</em></td>
<td>83</td>
</tr>
<tr>
<td><em>S.cerevisiae</em></td>
<td><em>S.cerevisiae</em></td>
<td>68</td>
</tr>
<tr>
<td><em>H.sapiens</em></td>
<td><em>H.sapiens</em></td>
<td>74</td>
</tr>
</tbody>
</table>

\(^a\)The percentage of the target organism transcripts that are covered by the design optimized using the design organism dataset.  
\(^b\)Assembly 3 (Galagan *et al.*, 2003).  

reduction), plant 374 pools (27% reduction) and human 329 pools (36% reduction). On the other hand, in the human dataset the greedy design covers more transcripts with 3 enzyme pairs and 512 pools than the fixed design with 4 enzyme pairs and the same number of pools (Fig. 3).

Table 4 shows that the design has been optimized using sequences from a closely related organism (*A.nidulans* and *N.crassa*), the coverage of the design is almost as good as using sequences from the organism itself. As expected, the designs optimized using a distant sequence set of different size (*H.sapiens*) perform worse.

A computer simulation can never precisely predict the transcript coverage of a cDNA-AFLP experiment. Our knowledge of the transcriptomes is incomplete which is the very reason to use cDNA-AFLP. Especially, the 3′-UTRs that are important when simulating the 3′-variant are often poorly characterized. Even if we would know the whole transcriptome, the coverage also depends on the number of different expressed transcripts in the experiment, one of the things the experiment is supposed to measure. Note, however, that this problem is independent of the method used to design the experiment. For example, the human sequence dataset that we used in the simulations is certainly far from being the complete transcriptome but on the other hand it is unlikely that the whole human transcriptome would be expressed in a particular experiment.

The manual cutting of electrophoresis bands and the further analysis of fragments is a significant part of the workload in any
large-scale cDNA-AFLP experiment. Even though our model does not directly address the cost of this step, our method can help the researcher to choose the selective primers so that reasonable band separation is achieved. Consequently, less purification is needed to extract the fragments of interest.

Despite the uncertainties, the suggested in silico optimization of experiments together with expert knowledge and experience could result in considerably more efficient use of resources in cDNA-AFLP experiments. The methods described give flexible means to explore different ways to allocate resources during the experimental design process.

ACKNOWLEDGEMENTS

The authors would like to thank Mr T. Mielikäinen for helpful discussions and Dr T. Pasanen for his comments on the manuscript. This work was supported by the Academy of Finland (grants 201560 and 77304), the European Union Biotechnology Program BIO4CT96-0535, the European Union V Framework Program Eurofung QLK3-CT-1999-00729 and, the research program Life 2000 (Academy of Finland and National Technology Agency of Finland, 2000-2003, Project no. 50023).

SUPPLEMENTARY DATA

Supplementary data for this paper are available on Bioinformatics online.

REFERENCES


