# A Multistart Constructive Heuristic for Sequencing by Hybridization Using Adaptive Memory

Eraldo L.R. Fernandes [1]

*Department of Computer Science, Catholic University of Rio de Janeiro, Rua Marquês de São Vicente 225, Rio de Janeiro, RJ 22453-900, Brazil.*

Celso C. Ribeiro [2]

*Department of Computer Science, Universidade Federal Fluminense, Rua Passo da Pátria 156, Niterói, RJ 24210-240, Brazil.*

**Abstract**

We propose a new multistart constructive heuristic for sequencing by hybridization, based on learning from an adaptive memory. Computational results show that this heuristic obtains systematically better solutions than more involving and time consuming techniques such as tabu search and genetic algorithms. The memory-based strategy is able to significantly improve the performance of memoryless construction procedures and is particularly suitable for scheduling problems whose best solutions are in most cases built by blocks of elements that appear together very often.

*Keywords:* Heuristics, multistart, adaptive memory, learning, sequencing

[1] Email: `eraldoluis@inf.puc-rio.br`
[2] Email: `celso@ic.uff.br`

## 1 Problem formulation

A DNA molecule can be viewed as a word in the alphabet {A,C,G,T} of nucleotides. The problem of DNA sequencing consists in determining the sequence of nucleotides that form a DNA molecule. There are currently two techniques for sequencing medium-size DNA molecules: gel electrophoresis and the chemical method. The approach of *sequencing by hybridization* (SBH) offers an interesting alternative to those above [6,7].

A short sequence of nucleotides is named an *oligonucleotide*. The set of all possible oligonucleotides of length $\ell$ is the *library* $C^\ell$. In this context, an oligonucleotide is also called a *probe*. The core of SBH is the hybridization experiment [1], which consists in identifying all oligonucleotides of a given length $\ell$ appearing in a DNA fragment of length $n$. This subset of probes, written as words of length $\ell$ over the alphabet {A,C,G,T}, is the *spectrum* of the target fragment. The spectrum of fragment ATAGGCAGGA in $C^4$ is formed by probes ATAG, TAGG, AGGC, GGCA, GCAG, CAGG, and AGGA.

The computational phase of SBH consists in using the spectrum to reconstruct the original fragment. The latter may be viewed as a sequence formed by all $n - \ell + 1$ probes in the spectrum, in which the last $\ell - 1$ letters of each probe coincide with the first $\ell - 1$ letters of the next. Two types of errors may be introduced along the hybridization experiment. *False positives* are probes that appear in the spectrum, but not in the target fragment. *False negatives* are probes that should appear in the spectrum, but do not. Errors in the spectrum make the reconstruction problem NP-hard [3]. The problem of sequencing by hybridization (SBH) is formulated as follows: given a spectrum $S$, the probe length $\ell$, the size $n$ and the first probe $v_0$ of the original fragment, find a sequence with length smaller than or equal to $n$ containing a maximum number of probes.

The maximization of the number of probes of the spectrum corresponds to the minimization of the number of errors in the solution. An instance of SBH may be represented by a directed weighted graph. There is a vertex for each probe in the spectrum. The weight of the edge connecting the vertices representing probes $u$ and $v$ is given by $w(u, v) = \ell - o(u, v)$, where $o(u, v)$ is the size of the largest sequence that is both a suffix of $u$ and a prefix of $v$. SBH consists in finding a simple path emanating from vertex $v_0$ in this graph, with a maximum number of vertices and weight smaller than or equal to $n-\ell$.

An example in which the spectrum is free of errors is given in Figure 1 (a). The solution is a path visiting all vertices and using only unit weight edges. The example in Figure 1 (b) depicts a situation in which probe GGCA was erroneously replaced by probe GGCG, introducing one false positive and one false negative error. The new optimal solution does not visit all vertices (due to the false positive) and uses one edge with weight equal to 2 (due to the false negative).

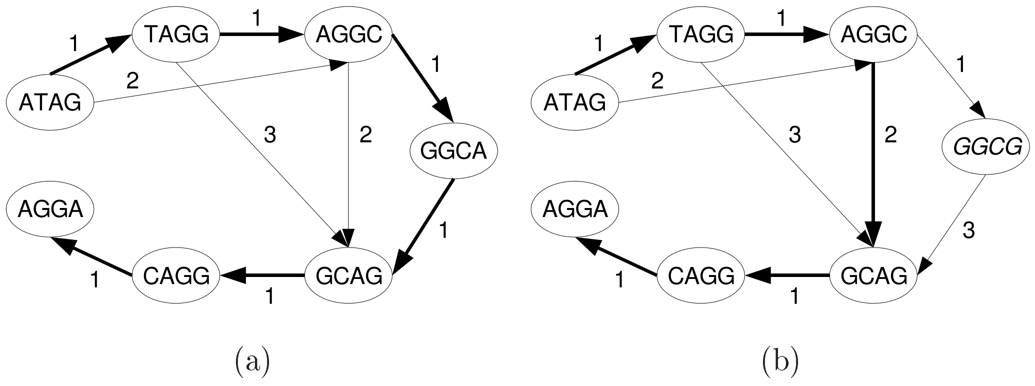


Fig. 1. Graphs and solutions for target fragment ATAGGCAGGA with $n = 10$ and $\ell = 4$: (a) no errors in the spectrum, (b) one false positive error (GGCG) and one false negative error (GGCA) in the spectrum.

Heuristics for SBH, handling both false positive and false negative errors, were proposed in [1,2,4]. We propose in the next section a memory-based greedy randomized algorithm for SBH, also handling both false positive and false negative errors. This algorithm makes use of a memory structure based on the frequency of consecutive probes appearing together in a set of elite solutions. Computational results illustrating the effectiveness of the new memory-based heuristic are reported in Section 3. Concluding remarks are made in the final section.

## 2 Memory-based multistart constructive heuristic

The multistart heuristic based on adaptive memory builds multiple solutions using a greedy randomized algorithm. An adaptive memory strategy based on the set of elite solutions previously found along the search is used to guide each iteration of the multistart heuristic [5].

The greedy algorithm iteratively extends a path emanating from the initial probe $v_0$ and currently formed by an ordered subset $P$ of the spectrum $S$. At each iteration, a new probe is appended at the end of the path defined by $P$ (or, simply, the path $P$). A probe $s \in S \setminus P$ is randomly selected from a restricted candidate list $R = \{v \in S \setminus P \mid o(u, v) \geq (1 - \alpha) \cdot \max_{t \in S \setminus P} o(u, t)$ and $w(P) + w(u, v) \leq n - \ell\}$, where $u$ is the last probe inserted into $P$, $0 \leq \alpha \leq 1$ is a parameter, and $w(P)$ is the weight of the path defined by $P$. List $R$ contains probes with a predefined minimum superposition with the last probe in $P$, restricting the search to more promising regions of the solution space. The construction of the solution stops when $R$ turns up to be empty.

A probe $v$ is selected from the restricted candidate list $R$ with a probability proportional to $E(u, v) = \lambda \cdot \text{value}(u, v) + \text{intensity}(u, v)$, where $u$ is the last probe inserted into $P$, $\text{value}(u, v) = \min_{t \in S \setminus P}\{w(u, t)/w(u, v)\}$ is higher when the superposition between probes $u$ and $v$ is stronger, and $\text{intensity}(u, v)$ is associated with the frequency in which edge $(u, v)$ appears in the memory defined by a set of elite solutions. A parameter $\lambda$ is used to balance the two criteria. The value of $\lambda$ should be high in the beginning of the algorithm, when the information in the memory is still weak.

We denote by $|P|$ the number of probes in a path $P$. The memory is formed by a set $Q$ of at most $q$ elite solutions found along the previous iterations. A new solution $P$ is a candidate to be inserted into the pool if $|P| > \min_{P' \in Q} |P'|$. This solution replaces the worst in the pool if $|P| > \max_{P' \in Q} |P'|$ (i.e., $P$ is better than the best solution currently in the pool) or if $\min_{P' \in Q} dist(P, P') \geq d$, where $d$ is a parameter of the algorithm and $dist(P, P')$ is the number of probes with different successors in $P$ and $P'$ (i.e., $P$ is better than the worst solution currently in the pool and sufficiently different from every other solution in the pool).

The contribution of the memory to the probability of selecting probe $v$ after probe $u$ is given by $\text{intensity}(u, v) = \sum_{P'' \in Q | (u,v) \in P''} \{|P''| / \max_{P' \in Q} |P'|\}$. We sketch in Figure 2 the pseudo-code with the main steps of the multistart constructive heuristic using adaptive memory.

## 3 Numerical results

The multistart constructive greedy randomized heuristic was implemented in `C++`, using version 3.3.2 of the `gcc` compiler. The `rand` function was used for the generation of pseudo-random numbers. The computational experiments were performed on a 2.4 GHz Pentium IV machine with 512 MB of RAM.

Test instances have been randomly generated from 40 sequences of human DNA obtained from GenBank, as described in [2]. Strings of length 109, 209, 309, 409, and 509 have been extracted from the GenBank source sequences. Next, a hybridization experiment was simulated by extracting probes of length $\ell = 10$ from each string, to obtain spectra with 100, 200, 300, 400, and 500 elements. Finally, some random probes were inserted in and extracted from the ideal spectra, generating instances with 20% of false positive errors and

```
procedure MultistartConstructiveHeuristic(S, v_0, ℓ, n)
    P* ← null;
    Set Q ← ∅;
    for i = 1 to N do
        Set P ← {v_0};
        Build the restricted candidate list R;
        while R ≠ ∅ do
            Compute the selection probability for each probe v ∈ R;
            Randomly select a probe v ∈ R;
            Extend the current solution P by appending v to its end;
            Update the restricted candidate list R;
        end;
        Use P to update the pool of elite solutions Q;
        if |P| > |P*| then set P* ← P;
    end;
    return P*;
end;
```

Fig. 2. Multistart constructive greedy randomized heuristic.

20% of false negative errors. A total of 200 instances have been created.

Preliminary computational experiments have been performed to tune the main parameters of the algorithm. The following settings were selected: $N = 10 \cdot |S|$ (number of iterations performed by the multistart heuristic), $\alpha = 0.2$ (weight parameter for the computation of the selection probabilities), $q = 0.04 \cdot |S|$ (size of the pool of elite solutions), and $d = 0.1 \cdot |S|$ (minimum difference for a solution to be accepted in the pool). Parameter $\lambda$ is set as $\lambda = 100 \cdot q$ in the first $|S|/2$ iterations, reduced to $\lambda = 10 \cdot q$ after the next $|S|/2$ iterations, and then reduced by $q$ after each $|S|/2$ iterations, until it is made equal to zero.

The memory-based multistart constructive heuristic (MCH) was compared with the tabu search algorithm (TS) in [2], the overlapping windows heuristic (OW) in [1], and the genetic algorithm (GA) in [4]. To evaluate the quality of the solutions found, we performed the alignment of the solution found by each algorithm with the original sequence. The higher is the alignment, the better is the corresponding solution. The numerical results are summarized in Tables 1 and 2, which depict the average alignment values and the average computation times observed for each algorithm over the 40 instances with the same size.

| | Sequence length | | | | |
|---|---|---|---|---|---|
| | 109 | 209 | 309 | 409 | 509 |
| TS | 105.1 | 184.5 | 244.6 | 315.1 | 312.3 |
| OW | 107.8 | 188.8 | 282.3 | 350.0 | 408.0 |
| GA | 107.8 | 206.2 | 296.3 | 401.1 | 498.0 |
| MCH | 109.0 | 208.6 | 305.5 | 402.6 | 504.2 |

Table 1
Average alignment values.

We first notice that the new memory-based multistart constructive heuristic (MCH) found much better solutions than the other heuristics. The alignments observed for the solutions produced by this algorithm are systematically higher. Furthermore, MCH was able to reconstruct the 40 original sequences of size 109 and 39 out of the 40 instances of sizes 209, 309, 409, and 509, corresponding to a total of 196 out of the 200 test instances. The overlapping windows heuristic found only 96 of the original sequences, while the tabu search heuristic identified only 88 sequences. The new heuristic is also faster than the others (the results displayed for the overlapping windows heuristic were obtained on a CRAY T3E-900 supercomputer).

| | Sequence length | | | | |
|---|---|---|---|---|---|
| | 109 | 209 | 309 | 409 | 509 |
| TS | <1.0 | 5.0 | 14.0 | 28.0 | 51.0 |
| OW | <1.0 | <1.0 | <1.0 | <1.0 | <1.0 |
| GA | 0.1 | 0.3 | 0.9 | 1.5 | 2.1 |
| MCH | 0.1 | 0.4 | 0.8 | 1.6 | 3.0 |

Table 2
Average computation times.

## 4 Concluding remarks

We proposed a new multistart constructive heuristic for sequencing by hybridization, based on learning from an adaptive memory. The adaptive memory strategy makes use of a set of elite solutions previously found along the search. The choice of the new element to be inserted into the partial solution at each iteration of the constructive procedure is based not only on greedy information, but also on frequency information extracted from the memory.

Computational results showed that the new heuristic obtains better results, both in terms of solution quality and computation times, than more involving and time consuming techniques such as tabu search and genetic algorithms. This memory-based strategy is able to significantly improve the performance of memoryless construction procedures and seems to be particularly suitable for scheduling problems whose best solutions are in most cases built by blocks of elements that appear together very often.

## References

[1] J. Blazewicz, P. Formanowicz, F. Guinand, and M. Kasprzak, *A heuristic managing errors for DNA sequencing*, Bioinformatics **18** (2002), 652–660.

[2] J. Blazewicz, P. Formanowicz, M. Kasprzak, W.T. Markiewicz, and T. Weglarz, *Tabu search for DNA sequencing with false negatives and false positives*, European Journal of Operational Research **125** (2000), 257–265.

[3] J. Blazewicz and M. Kasprzak, *Complexity of DNA sequencing by hybridization*, Theoretical Computer Science **290** (2003), 1459–1473.

[4] T.A. Endo, *Probabilistic nucleotide assembling method for sequencing by hybridization*, Bioinformatics **20** (2004), 2181–2188.

[5] C. Fleurent and F. Glover, *Improved construtive multistart strategies for the quadratic assignment problem using adaptive memory*, INFORMS Journal on Computing **11** (1999), 198–204.

[6] P.A. Pevzner, "Computational Molecular Biology: An Algorithmic Approach", The MIT Press, 2000.

[7] M.S. Waterman, "Introduction to Computational Biology: Maps, Sequences and Genomes", Chapman & Hall, 1995.