Global Optimization Methods for Genome Scaffolding

Sebastien François, Rumen Andonov, Hristo Djidjev, Dominique Lavenier

Global Optimization Methods for Genome Scaffolding

Sebastien François[1], Rumen Andonov[1], Hristo Djidjev[2], and Dominique Lavenier[1]

[1] IRISA/INRIA, Rennes, France
[2] Los Alamos National Laboratory, Los Alamos, NM 87545, USA

Abstract. We develop a method for solving genome scaffolding as a problem of finding a long simple path in a graph defined by the contigs that satisfies additional constraints encoding the insert-size information. Then we solve the resulting mixed integer linear program to optimality using the Gurobi solver. We test our algorithm on several chloroplast genomes and show that it is fast and outperforms other widely-used assembly algorithms by the accuracy of the results.

Keywords: genome assembly, scaffolding, contig, MILP, longest path problem, Gurobi

1 Introduction

High throughput sequencing (HTS) technologies provide millions of short genome fragments, called *reads*, which need to be assembled into a genome of interest. Today, most de-novo assemblers are based on de Bruijn graphs [12], whose vertices are all k-mers (k-length subwords of the reads) and whose edges connect all pairs of k-mers that share $k-1$ consecutive characters. Genomes can then be sought as maximal unambiguous paths in the de Bruijn graph. However, complex regions of the genome (i.e. regions with many repeats) generally fail to be assembled by this technique: if there are repeats (identical subregions of the genome) longer than the size of the reads, the entire genome cannot be built in a unique way. Various heuristics are used to bypass simple repeats, but they do not guarantee correct solutions. Hence, producing a full genome consists of several steps, namely assembly, scaffolding, and finishing, where the first step generates a list of *contigs* that represent the easy assembly regions of the genome.

In the second step, *scaffolding*, which is the focus of this paper, reads are linked together into *scaffolds*, which consist of sequence of contigs that overlap or are separated by gaps of given length. These gaps are generated during sequencing based on *paired-end* or *mate pair* reads [14,10]. These special reads can be represented as couples of fragments separated by a known distance (called *insert size*). They bring a long distance information that can be used for connecting contigs generated by de-Bruijn graphs. Scaffolding, which uses the set of contigs found during assembly and the insert-size information, is a very challenging problem, whose difficulties are rooted in technology imperfections including:

– Insert sizes are not precise. The technology provides approximate distance information only.
– For short contigs, which often represent short repeat regions, multiplicity cannot be precisely determined. This information is given by analyzing the coverage. Shorter the contig, worse the estimation.
– There may be erroneous contigs. Heuristics implemented for generating contigs may lead to chimeric sequences that wrongly connect two regions of the genome.

The difference between a contig and a scaffold is that a scaffold can contain regions that have not been completely solved. For example, for two contigs that have been unambiguously linked, the nucleotides sequence between them may have not been determined due to sequencing problem, or very high structure complexity. A last step (*finishing*) is generally required to enhance the scaffold. Additional information, such as sequences of close species, can be exploited.

This paper focuses on scaffolding only. Given a set of contigs and their relationships in terms of distances between them (insert sizes), we propose an optimization-based approach for finding the genome sequence as the longest sequence that is consistent with the given contig and linkage information. Specifically, we define a graph, which we call *contig graph*, whose vertices are the contigs and whose edges connect pairs of contigs that either overlap, or have a gap of size given by the insert-size information. Edges have weights that encode the corresponding distance information between the contigs and are negative in the case of overlaps and positive in the case of gaps. Vertices have weights equal to the lengths of the contigs they represent. Contigs with repeat factor s are represented as a set of s vertices with the same sets of neighbors. The length of a path in the resulting graph is defined as the sum of the weights of the vertices and edges in it. The scaffolding problem is reduced to finding a longest simple path such that as many as possible mate-pairs distances are satisfied (we call hereafter such path just a *longest path*). Since both conditions cannot generally be simultaneously satisfied, our objective function is a linear combination of them.

Unlike the shortest path problem, the longest path problem is NP-hard [3], which means that no polynomial time solution is likely to exist for it. We solve this problem by reformulating it as a mixed integer linear program (MILP) and developing a method that can solve the resulting optimization problem on genomes of up to 83 contigs and up to 900 binary variables. We analyze the performance of the algorithm on several chloroplast genomes and compare it to other scaffolding algorithms.

Most previous work on scaffolding is heuristics based, e.g., SSPACE [1], GRASS [4], and BESST [13]. Such algorithms may find in some cases good solutions, but their accuracies cannot be guaranteed or predicted. In contrast, our method always finds a longest path in the contig graph. There is no guarantee that the genome sequence corresponds to a longest path, but our experiments show that that is the case in many instances or, if not, there is a very small difference between the two. Exact algorithms for the scaffolding problem are

presented in [15], but the focus of that work is on finding structural properties of the contig graph that will make the optimization problem of polynomial complexity. In [11], integer linear programming is used to model the scaffolding problem, with an objective to maximize the number of links that are satisfied. In contrast, our objective is to maximize the length of the resulting scaffold. Moreover, we aim at producing a single path or cycle, rather than a set of paths and cycles. We believe that by requiring our solution to be a single path we avoid the risk of producing a set of paths for which the objective function is of high value, but which are inconsistent with a single path ordering. While our focus is on accuracy, [11] focuses on efficiency, and indeed their algorithm is faster than ours.

The contributions of this study are as follows:

- Our modeling of the scaffolding problem allows to solve *simultaneously* the set of subtasks specific for this problem like: contigs orientation and ordering, repeats, gap filling and scaffold extension.
- The scaffolding problem is reduced to finding a longest paths in a particular graph. In addition, these paths need to satisfy a set of distances between couples of vertices along these paths. We are not aware of previous approaches on scaffolding based on the longest path problem.
- We formulate the above problem as a mixed integer linear program (MILP) with several interesting properties like: cycles elimination constraints, using binary variables for the edges of the graph only. Vertices are modeled with real variables, but we prove that the integrality of these variables follows from other constraints.
- We tested this model on a set of chloroplast and bacteria genome data and showed that it allows to assemble the complete genome as a single scaffold. None of the publicly available scaffolding tools that we have tested targets single scaffolds (this is corroborated by the obtained numerical results).
- Our numerical experiments indicate that the relaxation of the mixed integer model is tight and produces upper bounds of excellent quality. This suggests a promising direction of research towards the scalability of our approach.

In the next Section 2 we describe our graph model and the formulation of the optimization problem and in Section 3 we present experimental results and comparison with other algorithms.

2 Modeling the scaffolding problem

2.1 Graph Modeling

We model the problem of scaffolding as path finding in a directed graph $G = (V, E)$ that we call a contig graph, where both vertices V and edges E are weighted. The set of vertices V is generated based on the set C of the contigs according the following rules: the contig i is represented by at least two vertices v_i and v_i' (forward/inverse orientation respectively). If the contig i is repeated k_i times, it generates $2k_i$ vertices. Denote $N = \sum_{i \in C} k_i$, therefore $|V| = 2N$.

The edges are generated following given patterns—a set of known overlaps/distances between the contigs. Any edge is given in the graph G in its forward/inverse orientation. We denote by e_{ij} the edge joining vertices v_i and v_j and the inverse of edge e_{ij} is $e_{j'i'}$. For any i, the weight w_i on a vertex v_i corresponds to the length of the contig i, while the weight l_{ij} on the edge e_{ij} corresponds to the value of the overlap/distance between contigs i and j. The problem then is to find a path in the graph G such that the total length (the sum over the traversed vertices and edges) is maximized, while a set of additional constraints are also satisfied:

- For any i, either vertex v_i or v_i' is visited (participates in the path).
- The orientations of the nodes does not contradict the constraints imposed by the mate-pairs. This is at least partially enforced by the construction of the graph.

To any edge $e \in E$ we associate a variable x_e. Its value is set to 1, if the corresponding edge participates in the assembled genome sequence (the associated path in our case), otherwise its value is set to 0. There are two kinds of edges: edges corresponding to overlaps between contigs, denote them by O (from overlaps), and edges associated with mate-pairs relationships, denote them by L (from links). We therefore have $E = L \cup O$. Let l_e be the length of the edge e. We have $l_e < 0 \ \ \forall e \in O$ and $l_e > 0 \ \forall e \in L$. Let w_v be the length of the contig corresponding to vertex v and denote $W = \sum_{v \in V} w_v$.

Let $A^+(v) \subset E$ (resp. $A^-(v) \subset E$) denote the subset of arcs in E leaving (resp. entering) node v.

2.2 Integer Linear Programming Formulation

We associate a binary variable for any edge of the graph, i.e.

$$\forall e \in O : x_e \in \{0,1\} \text{ and } \forall e \in L : g_e \in \{0,1\}. \tag{1}$$

Furthermore, to any vertex $v \in V$ we associate three variables, i_v, s_v, and t_v, which stand respectively for intermediate, source, and target for some path, and satisfy

$$0 \leq i_v \leq 1, \ \ 0 \leq s_v \leq 1, \ 0 \leq t_v \leq 1. \tag{2}$$

All three variables are set to zero when the associated vertex v participates in none of the paths. Otherwise, it could be either a source/initial (noted by $s_v = 1, t_v = 0, i_v = 0$), or a target/final ($t_v = 1, s_v = 0, i_v = 0$), or an intermediate vertex, in which case the equalities $i_v = 1, t_v = 0$ and $s_v = 0$ hold. Moreover, each vertex (or its inverse) can be visited at most once, i.e.

$$\forall (v, v') : i_v + i_{v'} + s_v + s_{v'} + t_v + t_{v'} \leq 1. \tag{3}$$

The four possibles states for a vertex v (to belong to none of the paths, or otherwise, to be a source, a target, or an intermediate vertex in some path) are provided by the following two constraints

$$s_v + i_v = \sum_{e \in A^+(v)} x_e \leq 1 \tag{4}$$

and

$$t_v + i_v = \sum_{e \in A^-(v)} x_e \leq 1. \tag{5}$$

Finally, only one sequence (a single path) is searched for

$$\sum_{v \in V} s_v = 1 \text{ and } \sum_{v \in V} t_v = 1. \tag{6}$$

Theorem 1. *The real variables $i_v, s_v, t_v, \forall v \in V$ take binary values.*

Proof. Let us analyse the four possibles cases deduced from (4) and (5). Denote $S^+ = \sum_{e \in A^+(v)} x_e$ and $S^- = \sum_{e \in A^-(v)} x_e$.

i) $S^+ = 0$ and $S^- = 0$.
In this case it follows from (2) that $s_v = i_v = t_v = 0$.

ii) $S^+ = 1$ and $S^- = 1$.
The above is equivalent to $s_v + i_v = 1$ and $t_v + i_v = 1$, which leads to $s_v = t_v$. Moreover, from (3) we have $s_v = t_v = 0$ and $i_v = 1$.

iii) $S^+ = 1$ and $S^- = 0$.
The above is equivalent to $s_v + i_v = 1$ and $t_v + i_v = 0$, which leads to $s_v - t_v = 1$. Hence, from (2), we have $t_v = 0$ and therefore $s_v = 1$ and $i_v = 0$.

iv) $S^+ = 0$ and $S^- = 1$.
This case is analogous to iii) and we have $s_v = i_v = 0$ and $t_v = 1$. □

We introduce a continuous variable $f_e \in R^+$ to express the quantity of the flow circulating along the arc $e \in E$

$$\forall e \in E : 0 \leq f_e \leq W. \tag{7}$$

For $e \in O$, the value of x_e is set to 1, if the arc e carries some flow and 0, otherwise. In other words, no flow can use the arc e when $x_e = 0$ as ensured by constraint

$$f_e \leq W x_e \quad \forall e \in O. \tag{8}$$

We use the flows f_e in the following constraints

$$\forall v \in V : \sum_{e \in A^-(v)} f_e - \sum_{e \in A^+(v)} f_e \geq i_v(w_v + \sum_{e \in A^-(v)} l_e x_e) - W s_v \tag{9}$$

$$Ws_v \leq \sum_{e \in A^+(v)} f_e. \tag{10}$$

The purpose of the last two constraints is manifold. When a vertex v is a source ($s_v = 1$), (9) and (10) generate and output from it an initial flow of sufficiently big value (W is enough in our case). When v is an intermediate vertex ($i_v = 1$), constraint (9) forces the flow to decrease by at least $l_{(u,v)} + w_v$ units when it moves form vertex u to its adjacent vertex v. The value of the flow thus is decreasing and this feature forbids cycles in the context of (4) and (5). When v is a final vertex, (9) is simply a valid inequality since the initial flow value is big enough.

We furthermore observe that because of (5), the constraint (9) can be written as follows

$$\forall v \in V : \sum_{e \in A^-(v)} f_e - \sum_{e \in A^+(v)} f_e \geq i_v w_v + \sum_{e \in A^-(v)} l_e x_e - W s_v. \tag{11}$$

The constraint (11) is linear and we keep it in our model instead of (9).

Furthermore, binary variables g_e are associated with links. For $(s,t) \in L$, the value of $g_{(s,t)}$ is set to 1 only if both vertices s and t belong to the selected path and the length of the considered path between them is in the given interval $[\underline{L}_{(s,t)}, \overline{L}_{(s,t)}]$. Constraints related to links are :

$$g_{(s,t)} \leq s_s + i_s + t_s \text{ and } g_{(s,t)} \leq s_t + i_t + t_t \tag{12}$$

as well as

$$\forall (s,t) \in L : \sum_{e \in A^+(s)} f_e - \sum_{e \in A^-(t)} f_e \geq \underline{L}_{(s,t)} g_{(s,t)} - M(1 - g_{(s,t)}) \tag{13}$$

$$\forall (s,t) \in L : \sum_{e \in A^+(s)} f_e - \sum_{e \in A^-(t)} f_e \leq \overline{L}_{(s,t)} g_{(s,t)}) + M(1 - g_{(s,t)}), \tag{14}$$

where M is some big constant.

We search for a long path in the graph and such that as much as possible mate-paired distances are satisfied. The objective hence is :

$$\max(\sum_{e \in O} x_e l_e + \sum_{v \in V} w_v(i_v + s_v + t_v) + p \sum_{e \in L} g_e) \tag{15}$$

where p is a parameter to be chosen as appropriate (currently $p = 1$).

Remark: Note that omitting constraints (6) from the above model generates a set of paths that cover "optimally" the contig graph, rather than a single path. We have tested this variant of the model, but the obtained solutions were too much fragmented and of worse quality compared to the single-path model.

3 Computational results

Here we present the results obtained on a small set of chloroplast and bacteria genomes given in Table 1. Synthetic sequencing reads have been generated for these instances applying ART simulator [7]. For the read assembly step required to produce contigs we applied minia [2] with parameter `unitig` instead of `contig` (a unitig is a special kind of a high-confidence contig). Based on these unitigs, the overlaps between them, as well as the mate-pair distances, we generated a graph as explained in Section 2.1. The graph generated for the Atropa belladonna genome is given in Figure 1.

We compared our results with the results obtained by three of the most recent scaffolding tools– SSPACE [1], BESST [13], and Scaffmatch [9]. In order to evaluate the quality of the produced scaffolds, we applied the QUAST tool [5]. The results are shown on Table 2. We observe that our tool GST (from Genscale Scaffolding Tool) is the only one that consistently assembles the complete genome (an unique scaffold in #scaffolds column) with more than 98% (and in four cases at least 99.9%) correctly predicted genome fraction and zero misassembles.

Our results were obtained on a standard laptop (Intel(R) Core(TM) i3-4000M with 2 cores at 2.4 GHz and 8 GB of RAM), and using Gurobi [6] for solving the MILP models.

Datasets	Total length	#unitigs	#nodes	#edges	#mate-pairs
Acinetobacter	3 598 621	165	676	8344	4430
Wolbachia	1 080 084	100	452	7552	2972
Aethionema Cordifolium	154 167	83	166	898	600
Atropa belladonna	156 687	18	36	114	46
Angiopteris Evecta	153 901	16	32	144	74
Acorus Calamus	153 821	15	30	134	26

Table 1: Scaffolding datasets.

Our next computational experiments focussed on comparing various relaxations and other related formulations for the MILP model described in the previous section. Let us denote in the sequel by BR (Basic Real) the model defined by the linear constraints (1), (2), (3), (4), (5), (6), (7), (8), (11), (10), (12), (13), (14) and objective function (15). Let BB (from Basic Binary) denote the same model except that constraint (2) is substituted by its binary variant, i.e.

$$\forall v \in V : i_v \in \{0,1\} \text{ and } s_v \in \{0,1\} \text{ and } t_v \in \{0,1\}. \tag{16}$$

According to Theorem 1, the models BB and BR are equivalent in quality of the results. We are interested here in their running time behavior. We study

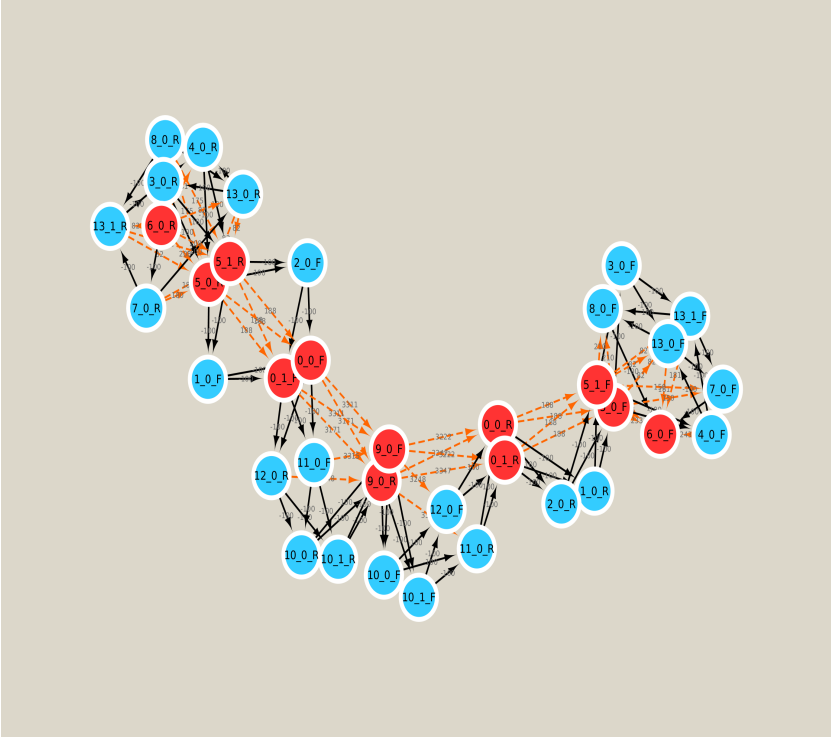

Fig. 1: The contig graph generated for the Atropa belladonna genome. Red/blue vertices correspond respectively to big/small contigs.

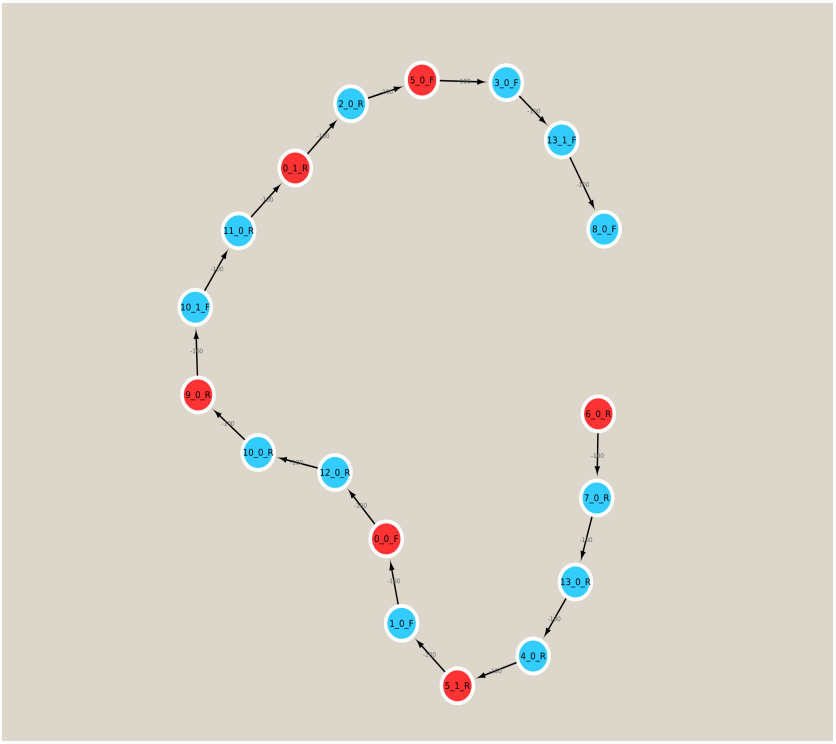

Fig. 2: The scaffold obtained for Atropa belladonna's genome shown on Figure 1.

Datasets	Scaffolder	Genome fraction	#scaffolds	# mis-assemblies	N's per 100 kbp
Acinetobacter	GST	98.536%	1	0	0
	SSPACE	98.563%	20	0	155.01
	BESST	98.539%	37	0	266.65
	Scaffmatch	98.675%	9	5	1579.12
Wolbachia	GST	98.943%	1	0	0
	SSPACE	97.700%	9	0	2036.75
	BESST	97.699%	49	0	642.90
	Scaffmatch	97.994%	2	2	3162.81
Aethionema Cordifolium	GST	100%	1	0	0
	SSPACE	95.550%	20	0	13603.00
	BESST	81.318%	30	0	1553.22
	Scaffmatch	82.608%	7	7	36892
Atropa belladonna	GST	99.987%	1	0	0
	SSPACE	83.389%	2	0	155.01
	BESST	83.353%	1	0	14.52
	Scaffmatch	83.516%	1	0	318.93
Angiopteris Evecta	GST	99.968%	1	0	0
	SSPACE	85.100%	4	0	0
	BESST	85.164%	2	0	1438.54
	Scaffmatch	85.684%	1	0	454.23
Acorus Calamus	GST	100%	1	0	0
	SSPACE	83.091%	4	0	126.39
	BESST	83.091%	4	0	127.95
	Scaffmatch	83.271%	1	1	3757.13

Table 2: Performance of different solvers on the scaffolding datasets from Table 1. GST is our tool.

as well the linear programming relaxation of BR (denoted by BR_{LP}) where the binary constraints (1) are substituted by

$$\forall e \in O : 0 \leq x_e \leq 1 \text{ and } \forall e \in L : 0 \leq g_e \leq 1. \tag{17}$$

Furthermore, let us relax in BR model all constraints related to mate-pairs distances (i.e. constraints (12, (13), (14)). We also omit from the objective function the last term that is associated to mate-pairs. Let us call this model LP (Longest Path) since it simply targets to find the longest path in the contig graph. Any solution of this model can be extended to a solution of BR model by a completion $g_e = 0 \ \forall e \in L$. Its optimal value yields a lower bound for the main model BR.

The results obtained by each one of these models are presented in Table 3. From these results we first observe that, as expected, the model BR outperforms BB in running time. With respect to the quality of the obtained results, the results with the relaxed models are very encouraging. The upper bounds computed by the linear relaxation BR_{LP} are extremely close to the exact values computed by BR model. Furthermore, the quality of the lower bound found by the longest path approach LP is also very good. Interestingly, we observed for the given instances that this value is close to the genome's size. We presume that the LP model can be used for predicting the length of the genome when it is unknown.

4 Conclusion

We developed and tested algorithms for scaffolding based on a version of the longest path problem and MILP representation. Our algorithms significantly outperform three of the best known scaffolding algorithms with respect to the quality of the scaffolds. Regardless of that, we consider the current results as a work in progress. The biggest challenge is to extend the method to much bigger genomes. We plan to use some additional ideas and careful implementation to increase the scalability of the methods without sacrificing the accuracy of the results.

References

1. Boetzer, M., Henkel, C.V., Jansen, H.J., Butler, D., Pirovano, W.: Scaffolding preassembled contigs using SSPACE. Bioinformatics (Oxford, England) 27(4), 578–579 (Feb 2011)
2. Chikhi, R., Rizk, G.: Space-efficient and exact de Bruijn graph representation based on a Bloom filter. In: WABI. Lecture Notes in Computer Science, vol. 7534, pp. 236–248. Springer (2012)
3. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA (1990)
4. Gritsenko, A.A., Nijkamp, J.F., Reinders, M.J., Ridder, D.d.: GRASS: a generic algorithm for scaffolding next-generation sequencing assemblies. Bioinformatics 28(11), 1429–1437 (2012), `http://bioinformatics.oxfordjournals.org/content/28/11/1429.abstract`

Datasets	Models	Obj	1^{st} term (length)	2^{nd} term (# satisfied links)	Time
Acinetobacter	BB	N/A	N/A	N/A	15m00.000s*
	BR	3 598 689	3 598 499	190	6m27.440s
	BR_{LP}	3 598 977	3 597 826	1151	0m44.508s
	LP	3 598 518	3 598 518	N/A	1m16.318s
Wolbachia	BB	N/A	N/A	N/A	15m00.000s*
	BR	1 075 949	1 075 856	93	3m13.144s
	BR_{LP}	1 076 053	1 075 624	429	0m25.428s
	LP	1 075 857	1 075 857	N/A	0m18.694s
Atropa belladonna	BB	156 501	156 488	13	0m1.151s
	BR	156 501	156 488	13	0m0.780s
	BR_{LP}	156 507	156 468	39	0m0.720s
	LP	156 488	156 488	N/A	0m0.296s
Angiopteris Evecta	BB	145 542	145 534	8	0m28.728s
	BR	145 542	145 534	8	0m1.084s
	BR_{LP}	145 556	145 501	55	0m0.752s
	LP	145 535	145 535	N/A	0m0.308s
Acorus Calamus	BB	153 976	153 970	6	0m1.343s
	BR	153 976	153 970	6	0m1.060s
	BR_{LP}	153 981	153 959	22	0m0.768s
	LP	153 970	153 970	N/A	0m0.261s
Aethionema Cordifolium	BB	151 563	151 445	118	15m00.000s*
	BR	151 570	151 445	125	15m00.000s*
	BR_{LP}	151 610	151 200	410	0m1.992s
	LP	151 445	151 445	N/A	0m1.548s

Table 3: Performance of the basic MILP model and some of its relaxations and related formulations on the scaffolding datasets from Table 1. The symbol $*$ indicates that the corresponding execution has been stopped by time limit.

5. Gurevich, A., Saveliev, V., Vyahhi, N., Tesler, G.: QUAST: quality assessment tool for genome assemblies. Bioinformatics (Oxford, England) 29(8), 1072–1075 (Apr 2013)
6. Gurobi Optimization, I.: Gurobi optimizer reference manual version 3.0. (2010)
7. Huang, W., Li, L., Myers, J.R., Marth, G.T.: Art: a next-generation sequencing read simulator. Bioinformatics 28(4), 593–594 (2012), `http://bioinformatics.oxfordjournals.org/content/28/4/593.abstract`
8. Kolodner, R., Tewari, K.K.: Inverted repeats in chloroplast DNA from higher plants. Proceedings of the National Academy of Sciences of the United States of America 76(1), 41–45 (01 1979), `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC382872/`
9. Mandric, I., Zelikovsky, A.: Scaffmatch: scaffolding algorithm based on maximum weight matching. Bioinformatics (2015), `http://bioinformatics.`

oxfordjournals.org/content/early/2015/05/14/bioinformatics.btv211.abstract
10. Medvedev, P., Pham, S., Chaisson, M., Tesler, G., Pevzner, P.: Paired de Bruijn graphs: A novel approach for incorporating mate pair information into genome assemblers. Journal of Computational Biology 18(11), 1625–1634 (11 2011), http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3216098/
11. Nicolas, B., Annie, C., Coletta, R., de Givry, S., Leleux, P., Thomas, S.: An integer linear programming approach for genome scaffolding. In: Workshop on Constraint based Methods for Bioinformatics (2015)
12. Pevzner, P.A., Tang, H., Waterman, M.S.: An Eulerian path approach to DNA fragment assembly. Proceedings of the National Academy of Sciences 98(17), 9748–9753 (2001), http://www.pnas.org/content/98/17/9748.abstract
13. Sahlin, K., Vezzi, F., Nystedt, B., Lundeberg, J., Arvestad, L.: BESST - efficient scaffolding of large fragmented assemblies. BMC Bioinformatics 15, 281 (2014)
14. Weber, J.L., Myers, E.W.: Human whole-genome shotgunsequencing. Genome Research 7(5), 401–409 (1997), http://genome.cshlp.org/content/7/5/401.short
15. Weller, M., Chateau, A., Giroudeau, R.: Exact approaches for scaffolding. BMC bioinformatics 16(Suppl 14), S2 (2015)