Global Optimization for Scaffolding and Completing Genome Assemblies

Sebastien François, Rumen Andonov, Dominique Lavenier, Hristo Djidjev

Global Optimization for Scaffolding and Completing Genome Assemblies

Sebastien François, Rumen Andonov, Dominique Lavenier, Hristo Djidjev

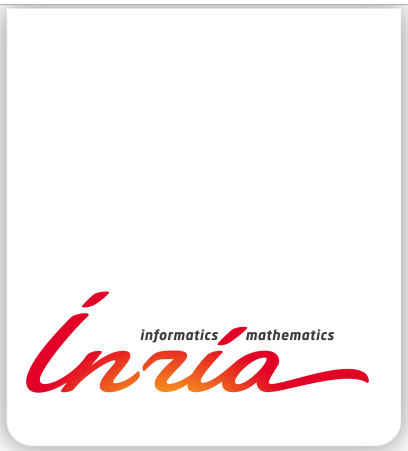
RESEARCH REPORT

N° 9050

March 2017

Project-Team GenScale

ISSN 0249-6399 ISRN INRIA/RR--9050--FR+ENG



Global Optimization for Scaffolding and Completing Genome Assemblies

Sebastien François*, Rumen Andonov†, Dominique Lavenier‡,
Hristo Djidjev§

Project-Team GenScale

Research Report n° 9050 — March 2017 — ?? pages

Preliminary version has been presented at the Workshop on Constraint-Based Methods for Bioinformatics 2016

* sefra35@gmail.com, IRISA/INRIA/University of Rennes 1, Rennes, France
† randonov@irisa.fr, IRISA/INRIA/University of Rennes 1, Rennes, France, corresponding author
‡ lavenier@irisa.fr, CNRS/INRIA, Rennes, France
§ djidjev@lanl.gov, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex

Abstract: This work focuses simultaneously on both the scaffolding and gap filling phases of de nouveau genome assembly. Given a set of contigs and their relationships–overlaps and/or remoteness in terms of distances between them–we propose an optimization-based approach for finding the genome sequence as a longest sequence that is consistent with the given contig and linkage information. Specifically, we define a graph, which we call a contig graph, that encodes information about contigs and overlaps and mate-pair distances between them, and reduce the scaffolding problem to the problem of finding a longest simple path in that graph such that as many as possible mate-pairs distances are satisfied. Since both conditions cannot generally be simultaneously satisfied, our objective function is a linear combination of the path length and penalties for distance mismatches. Unlike the shortest path problem with non-negative weights, for which efficient polynomial-time algorithms exist, the longest weighted path problem is NP-hard. We solve this problem by reformulating it as a mixed integer linear program (MILP) and develop a method that exactly solves the resulting program on genomes of up to 165 contigs and up to 6682 binary variables.

An advantage of our approach is that the modeling of scaffolding as a longest path problem allows one to solve simultaneously several subtasks specific for this problem like: contig orientation and ordering, repeats, gap filling, and scaffold extension, which in other approaches are targeted as separate problems. We are not aware of previous approaches on scaffolding based on the longest path problem reduction. A drawback of the typically used strategy of constructing a set of disjoint paths, rather than a single path, is that it would require additional steps of gap filling and scaffold extension, involving additional work. Moreover, it would make impossible to find a provably optimal final solution, since, even if each separate problem is implemented optimally, their combination may not be optimal. 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. Compared to the publicly available scaffolding tools that we have tested, our solution produces assemblies of significantly higher quality.

Key-words: genome assembly, scaffolding, contig, longest simple weighted path problem, integer programming

Optimisation globale appliquée à l'assemblage génomique

Résumé : Ce travail porte particulièrement sur les étapes de scaffolding et de gap-filling lors de l'assemblage de novo d'un génome. Etant donné un ensemble de contigs et leurs relations (chevauchement et/ou éloignement), nous proposons une approche basée sur l'optimisation, qui consiste à trouver le plus long assemblage possible consistant avec les distances attendues. Nous résolvons ce problème en le formulant sous la forme d'une programme mixte en nombres entiers (MILP). Notre méthode permet de résoudre exactement des instances comptant jusqu'à 6682 variables binaires pour 165 contigs.

L'un des avantages de notre modélisation est qu'elle permet de résoudre simultanément différentes sous taches spécifiques à l'assemblage complet d'un génome : déterminer l'orientation de chaque contig, ordonnancer les répétitions, compléter les scaffolds et les étendre en un assemblage unique etc... Ce qui constitue d'ordinaire un ensemble de problèmes à traiter séquentiellement. (Nécessitant alors des solutions spécifiques pour chacun d'entre eux). A notre connaissance, aucune autre approche ne propose de réduire le problème de l'assemblage génomique à celui de la recherche du plus long chemin.

Mots-clés : Pas de motclef

1 Introduction

Modern Next-Generation Sequencing (NGS) techniques are not able to output the whole genome sequence in one large sequence, but instead output billions of short DNA sequences, called *reads*. These reads are extremely redundant, erroneous, and, consequently, unusable practically. *Genome assembly* is the challenging computational task, consisting in reconstructing the full genome sequence from these fragmented raw data. This is a complex procedure, usually composed of three main steps: (1) generation of *contigs*, which are long consensual DNA sequences issued from the overlapping of the reads; (2) orientation and ordering of these contigs, called the *scaffolding*; (3) gap-filling and contig extension. A notable weakness of the contemporary approaches for *de-novo assembly* is to consider the above process as a set of independent tasks and not be able to propose a global optimal solution.

The first step generates a list of *contigs* (assembled ungapped sequences) that represent the "easily assembled regions" of the genome. Building contigs is currently supported by methods using a specific data structure called *de-Bruijn* graph [**?**].

Whereas the main challenge in the first step is in the sheer size of the raw data, in the second step, scaffolding, the data is of moderate size, but the problem remains largely open because of its NP-hard complexity [**?**]). The goal here is to provide a reliable order and orientation of the contigs in order to link them together into *scaffolds*– a sequence of contigs that overlap or are separated by gaps of a given length. These gaps are generated during sequencing based on *paired-end* or *mate pair* reads [**?**], and have distance information associated with them. Specifically, they can be represented as couples of fragments separated by a known distance (called *insert size*) and provide information about distance between contigs.

While the ultimate goal of the genome assembly is to generate a complete genome, the scaffolding phase usually produces a set of multiple scaffolds that, in addition, may contain inside them regions that have not been completely predicted. Further steps, such as *gap-filling* and *contig extension* are generally required to complete the genome.

This paper focuses on the scaffolding phase. Given a set of contigs and their relationships– overlaps and/or remoteness 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 contigs and linkage information. Specifically, we define a graph, which we call *contig graph*, that encodes information about contigs and distances between them, and reduce the scaffolding problem to finding a longest simple path in that graph 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 with non-negative weights, for which efficient polynomial-time algorithms exist, the longest weighted path problem is NP-hard [**?**], 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 develop a method that exactly solves the resulting program on genomes of up to 165 contigs and up to 6682 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 [**?**], GRASS [**?**], and BESST [**?**]. 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 [**?**],

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 [**?**], 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. While our focus is on accuracy, [**?**] focuses on efficiency, and indeed their algorithm, being a kind of heuristics, is faster than ours.

The contributions of this study are as follows:

- Our modeling of the scaffolding problem as a longest path problem allows to solve *simultaneously* the set of subtasks specific for this problem like: contigs orientation and ordering, repeats, gap filling and scaffold extension, which in other approaches are separate phases.
- 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. The disadvantage of having a set of disjoint paths, rather than a single path, is that it would require additional steps of gap filling and scaffold extension, involving additional work. Moreover, it would make it impossible to produce an optimal final solution, since, even if each separate problem is implemented optimally, their combination may not be optimal.
- 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. Moreover, the commonly used approach for solving the longest weighted simple path when the initial (source) and final (target) vertices are unknown consists in artificially adding these two vertices and $2|V|$ edges in the graph $G = (V, E)$ [**?**]. This increases by $2|V|$ the number of binary variables, which is a drawback when the density of the graph is small (as is the case of scaffolding graph). In contrast, our modeling does not require such a graph transformation and requires fewer binary variables.
- 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.

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. The values of k_i are computed in the contigs-generation phase. 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 mate-pairs. This is at least partially enforced by the construction of G.

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 $e = (u, v)$. We have $l_e < 0$ and $|l_e| < \min\{w(u), w(v)\}$, $\forall e \in O$, 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 Mixed 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

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

All three variables are set to zero when the associated vertex v participates in none of the paths. Otherwise, v can 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

$$s_v + i_v = \sum_{e \in A^+(v)} x_e, \quad t_v + i_v = \sum_{e \in A^-(v)} x_e. \tag{4}$$

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

$$\sum_{v \in V} s_v = 1 \text{ and } \sum_{v \in V} t_v = 1. \tag{5}$$

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

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

$$\forall e \in E : 0 \leq f_e \leq W x_e. \tag{6}$$

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 $x_e = 0$.

We use the flows f_e in the following constraints, $\forall v \in V$,

$$\sum_{e \in A^-(v)} f_e - \sum_{e \in A^+(v)} f_e \geq (i_v + t_v)(w_v + \sum_{e \in A^-(v)} l_e x_e) - W s_v, \quad W s_v \leq \sum_{e \in A^+(v)} f_e. \tag{7}$$

The purpose of the last two constraints is manifold. When a vertex v is a source ($s_v = 1$), (**??**) generates and outputs 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 (**??**) forces the flow to decrease by at least $l_{(u,v)} + w_v$ units when it moves from vertex u to its adjacent vertex v. The value of the flow thus is decreasing and this feature forbids cycles in the context of (**??**). When v is a final vertex, (**??**) is simply a valid inequality for the input flow.

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

$$\forall v \in V : \sum_{e \in A^-(v)} f_e - \sum_{e \in A^+(v)} f_e \geq (i_v + t_v) w_v + \sum_{e \in A^-(v)} l_e x_e - W s_v. \tag{8}$$

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

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 :

$$g_{(s,t)} \leq s_s + i_s + t_s \text{ and } g_{(s,t)} \leq s_t + i_t + t_t \tag{9}$$

$$\forall (s,t) \in L : \sum_{e \in A^+(s)} f_e - \sum_{e \in A^-(t)} f_e \geq \underline{L}_{(s,t)} g_{(s,t)} - M(1 - g_{(s,t)}) \tag{10}$$

$$\forall (s,t) \in L : \sum_{e \in A^+(s)} f_e - \sum_{e \in A^-(t)} f_e \leq \overline{L}_{(s,t)} g_{(s,t)}) + M(1 - g_{(s,t)}), \tag{11}$$

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 :

$$\max \left(\sum_{e \in O} x_e l_e + \sum_{v \in V} w_v (i_v + s_v + t_v) + p \sum_{e \in L} g_e \right) \tag{12}$$

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

3 Computational results

Here we present the results obtained on a small set of chloroplast and bacteria genomes given in Table **??**. Synthetic sequencing reads have been generated for these instances applying ART simulator [**?**]. For the read assembly step required to produce contigs we applied minia [**?**] with

Datasets	Total	#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.

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 **??**.

Our results were obtained on an Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz with 20 cores, 64 GB of RAM, and using Gurobi solver for solving the MILP models. We compared our results with the results obtained by three of the most recent scaffolding tools– SSPACE [**?**], BESST [**?**], and Scaffmatch [**?**]. In order to evaluate the quality of the produced scaffolds, we applied the QUAST tool [**?**]. The results are shown on Table **??**. 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 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 (**??**-**??**) and (**??**-**??**) and objective function (**??**). Let BB (from Basic Binary) denote the same model except that constraint (**??**) is substituted by its binary variant, i.e.

$$\forall v \in V : i_v \in \{0,1\} \text{ and } s_v \in \{0,1\} \text{ and } t_v \in \{0,1\}. \tag{13}$$

According to Theorem **??**, the models BB and BR are equivalent in quality of the results. We are interested here in their running time behavior. We study as well the linear programming relaxation of BR (denoted by BR_{LP}) where the binary constraints (**??**) are substituted by

$$\forall e \in O : 0 \leq x_e \leq 1 \text{ and } \forall e \in L : 0 \leq g_e \leq 1. \tag{14}$$

Furthermore, let us omit from the objective function the last term that is associated to mate-pairs, as well all constraints in BR model that relate to to mate-pairs distances (i.e. constraints (**??**, (**??**), (**??**)). We call this model LP (Longest Path) since it simply targets finding 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.

By experimentally analyzing these models (see Table **??** in the Appendix for the data) we observe that, as expected, the model BR significantly outperforms BB in running time. 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.

Datasets	Scaffolder	Genome fraction	#scaffolds	# misassemblies	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 datasets from Table **??**. GST is our tool.

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 without sacrificing the accuracy of the results.

References

[1] Boetzer, M., Henkel, C.V., Jansen, H.J., Butler, D., Pirovano, W.: Scaffolding pre-assembled contigs using SSPACE. Bioinformatics (Oxford, England) 27(4), 578–579 (Feb 2011)

[2] Bui, Q.T., Deville, Y., Pham, Q.D.: Exact methods for solving the elementary shortest and longest path problems. Annals of Operations Research 244(2), 313–348 (2016)

[3] 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)

[4] 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)

[5] 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)

[6] Gurevich, A., Saveliev, V., Vyahhi, N., Tesler, G.: QUAST: quality assessment tool for genome assemblies. Bioinformatics 29(8), 1072–1075 (Apr 2013)

[7] Huang, W., Li, L., Myers, J.R., Marth, G.T.: Art: a next-generation sequencing read simulator. Bioinformatics 28(4), 593–594 (2012)

[8] Huson, D.H., Reinert, K., Myers, E.W.: The greedy path-merging algorithm for contig scaffolding. J. ACM 49(5), 603–615 (2002)

[9] Mandric, I., Zelikovsky, A.: ScaffMatch: scaffolding algorithm based on maximum weight matching. Bioinformatics (2015)

[10] 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)

[11] Pevzner, P.A., Tang, H., Waterman, M.S.: An Eulerian path approach to DNA fragment assembly. PNAS 98(17), 9748–9753 (2001)

[12] Sahlin, K., Vezzi, F., Nystedt, B., Lundeberg, J., Arvestad, L.: BESST - efficient scaffolding of large fragmented assemblies. BMC Bioinformatics 15, 281 (2014)

[13] Weber, J.L., Myers, E.W.: Human whole-genome shotgun sequencing. Genome Research 7(5), 401–409 (1997)

[14] Weller, M., Chateau, A., Giroudeau, R.: Exact approaches for scaffolding. BMC bioinformatics 16(Suppl 14), S2 (2015)

Appendix

A Proof of Theorem ??

Proof. Let us analyse the four possibles cases deduced from (**??**). 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 (**??**) 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 (**??**) 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 (**??**), 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 $t_v = 1$.

□

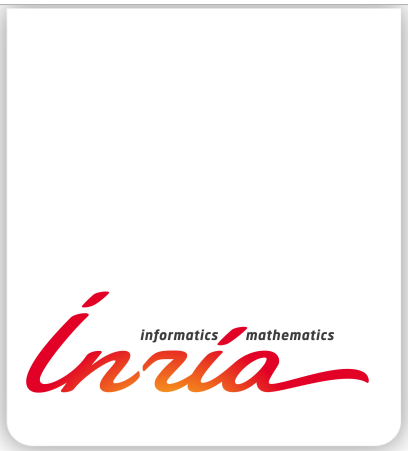
B Computational results - model comparison

B Data for MILP versions comparison

Datasets	Models	Obj	path term (length)	mate-pairs term (# satisfied links)	Time
Acinetobacter	BB	N/A	N/A	N/A	15m00.000s*
	BR	3 598 689	3 598 499	190	3m13.878s
	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}	1 076 109	1 075 857	252	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	3m15.534s
	BR_{LP}	151 610	151 200	410	0m1.992s
	LP	151 445	151 445	N/A	0m1.548s

Table 3: Performance of the basic model, some of its relaxations and related formulations. The symbol $*$ indicates that the execution has been stopped by time limit.

Contents



RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399