UNIVERSIDAD DE BUENOS AIRES
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

# REPRESENTACIONES MINIMALES DE GRAFOS DE INTERVALOS UNITARIOS

Tesis presentada para optar al título de
Licenciado de la Universidad de Buenos Aires
en el área Ciencias de la Computación

**Esteban Capillo**

Director de tesis: Dr. Francisco Soulignac

Buenos Aires, 2014

# Representaciones minimales de grafos de intervalos unitarios

En esta tesis estudiamos el problema de decidir si, dado un modelo de intervalos propios $\mathcal{I}$ y una longitud $\ell$, existe un modelo de intervalos unitario $\mathcal{U}$ de longitud $\ell$ equivalente a $\mathcal{I}$ cuyos extremos son enteros y satisfacen ciertas restricciones de separación. Como resultado de esta tesis, diseñamos un algoritmo que encuentra $\mathcal{U}$ en tiempo cuadrático, en caso que sea posible. Más aún, cuando dicho modelo unitario no existe, el algoritmo muestra una evidencia de este hecho. Asimismo, mostramos un algoritmo que encuentra el mínimo valor de $\ell$ para el cual el problema tiene solución, que requiere tiempo cuadrático. Por último, mostramos cómo puede aplicarse este algoritmo para encontrar los mínimos $k$ y $q$ tal que el grafo de intersección de $\mathcal{I}$ sea un subgrafo inducido de $P_q^k$.

**Palabras clave:** *modelo de intervalos unitarios, representación mínima, restricciones de separación, potencia de caminos.*

# Minimal representations of unit interval graphs

In this thesis we study the problem of deciding if, given a proper interval model $\mathcal{I}$ and a length $\ell$, there exists a unit interval model $\mathcal{U}$ of length $\ell$ equivalent to $\mathcal{I}$ whose extremes are integer and such that it satisfies certain separation constraints. Our main result is an algorithm that finds $\mathcal{U}$ in quadratic time, when it is possible. Moreover, when such an unit interval model does not exist, the algorithm outputs some evidence about this fact. We also show a quadratic time algorithm that finds the minimum value of $\ell$ for which the problem has a solution. Finally, we show how we can apply this algorithm to find the minimum $k$ and $q$ such that the intersection graph of $\mathcal{I}$ is an induced subgraph of $P_q^k$.

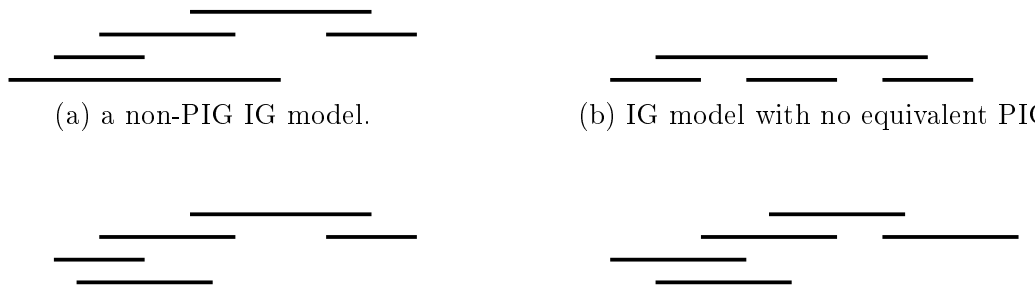
**Keywords:** *unit interval model, minimal representation, separation constraints, powers of paths*

Dedicated to Maria & Gelacio

# Contents

# 1 Introduction

Our goal in this thesis is to study the MINIMAL UIG REPRESENTATION (MINUIG) problem that consists in transforming a proper interval model into an equivalent unit interval model whose intervals are of minimum length with integer extremes. An *interval model* (IG) is simply a family $\mathcal{I}$ of open intervals on the real line. When no interval of $\mathcal{I}$ is properly contained in another interval, $\mathcal{I}$ is said to be a *proper* interval model (PIG). If, in addition, all the intervals have the same length, then $\mathcal{I}$ is a *unit* interval model (UIG). Figure 1.1 shows examples of IG, PIG, and UIG models.



(a) a non-PIG IG model.

(b) IG model with no equivalent PIG

(c) a PIG model equivalent to (a)

(d) a UIG model equivalent to (a)

**FIGURE 1.1** – *Examples of IG, PIG and UIG models. Note that a PIG graph may admit non-PIG models, as shown in (a) and (c).*

Each IG model $\mathcal{I}$ can be associated with a graph that represents the intersection of the intervals. The graph $G(\mathcal{I})$ has one vertex $v(I)$ for each $I \in \mathcal{I}$, and two vertices $v(I)$ and $v(I')$ of $G(\mathcal{I})$ are adjacent if and only if $I$ and $I'$ have a nonempty intersection. Model $\mathcal{I}$ is said to be a *model* or *representation* of $G(\mathcal{I})$, while $G(\mathcal{I})$ is said to *admit* $\mathcal{I}$. In general, a graph $G$ admits zero or infinite IG models (see Figures 1.1 and 1.2). However, much of these models are "equivalent", as they are obtained from each other by rescaling or moving the intervals. Formally, two models $\mathcal{I}$ and $\mathcal{I}'$ are *equivalent* when their extremes appear in the exact same order. By moving all the extremes of $\mathcal{I}$ by infinitesimal values, an equivalent IG model in which no two extremes coincide can be obtained. Hence, as it is customary in the literature (e.g. [7]), we assume that all the extremes of $\mathcal{I}$ are different.

Two famous and well studied classes of graphs are defined according to the kinds of interval models they admit. The class of *interval graphs* (IG) is formed by those graphs

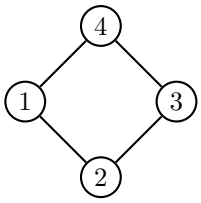



**Figure 1.2** – *Example of a graph admitting no IG models.*

that admit at least one IG model. Similarly, *proper interval graphs* (PIG) are those graphs that admit a PIG representation. Clearly, every PIG graph is an IG graph, but the converse is not always true; the interval graph $K_{1,3}$ admits no proper interval models (see Figure 1.1 (b)). Thus, the problem of transforming an IG model into an equivalent PIG model needs not have a solution. Similarly, the problem PIGtoUIG that asks for a UIG model equivalent to an input PIG model $\mathcal{I}$ is feasible if and only if $\mathcal{I}$ is equivalent to a UIG model. A priori, this needs not to be the case for every PIG model. However, Roberts [14] proved that every PIG model is equivalent to a UIG model, i.e., PIGtoUIG is well defined for every possible input. By Roberts' Theorem, the class of PIG graphs is also referred to as the class of UIG graph.

Interval orders are strongly related to interval graphs. A *partially ordered set (poset)* is a pair $(X, <)$ such that $<$ is an irreflexive, asymmetric and transitive relation on $X$. Each interval model $\mathcal{I}$ is associated with a special poset $P(\mathcal{I}) = (X, <)$ where $X$ has an element $x(I)$ for each $I \in \mathcal{I}$, and $x(s,t) < x(s',t')$ if and only if $t < s'$ for ever pair of open intervals $(s,t), (s',t') \in I$[1]. That is, $x(I) < x(I')$ when $I$ appears completely to the left of $I'$ on the real line. Model $\mathcal{I}$ is said to be a *model* or *representation* of $P(\mathcal{I})$, whereas $P(\mathcal{I})$ is said to *admit* $\mathcal{I}$. The relation between interval graphs and interval orders is quite obvious. There is an edge between $v(I)$ and $v(I')$ in $G(\mathcal{I})$ if and only if $x(I)$ and $x(I')$ are incomparable in $P(\mathcal{I})$. In technical words, $G(\mathcal{I})$ is the co-comparability graph of $P(\mathcal{I})$.

Semiorders are to UIG models what interval orders are to IG models. A *semiorder* is an interval order $P$ that admits an interval representation in which all intervals have the same length. In other words, a semiorder is just an interval order $P(\mathcal{I})$ represented by a UIG model $\mathcal{I}$. The seminal paper by Scott and Suppes [15] implies that $P(\mathcal{I})$ is a semiorder whenever $\mathcal{I}$ is a PIG model; this is somehow equivalent to Roberts' theorem [14]. Thus, it makes sense to consider the MinSO problem whose purpose is to find a UIG model with integer extremes that is of minimum length and represents $P(\mathcal{I})$, when the input is a PIG model $\mathcal{I}$. The MinSO problem is just a restatement of the MinUIG problem in terms of semiorders.

[1] For the sake of notation, we write $x(s,t)$ instead of $x((s,t))$ for an interval $(s,t)$.

Several authors have considered the PIGtoUIG and MinUIG problems. In the following section we review these algorithms in some detail. Before doing so, however, we find it enlightening to review the history of these algorithms and the motivations that led to them.

The treatment of the PIGtoUIG problem started with the seminal works by Scott and Suppes [15] and Roberts [14]. Scott and Suppes defined the axioms of semiorders and proved that a poset is a semiorder if and only if it contains no $2+2$ and no $1+3$ suborders. In graph theoretical terms, $2+2$ and $1+3$ are the equivalents of $C_4$ and $K_{1,3}$ graphs. Thus, any comparability graph with no induced $C_4$'s nor $K_{1,3}$ are UIG graphs. This fact was exploited by Roberts' to prove the equivalence between PIG graphs and UIG graphs. These seminal results do not tell us how to transform a PIG model into a UIG model; they just prove that this is possible.

In 1990, Pirlot considered the MinSO problem from a theoretical point of view in [13] (see also [12]). In his original formulation, the extremes of the UIG model constructed need not be integers. Instead, he considers the problem of finding a UIG model in which $t$ and $s'$ are separated by a threshold $\varepsilon \in \mathbb{R}_{>0}$ for every interval $(s,t) < (s',t')$. In particular, he proves that all the extremes are integers when $\varepsilon = 1$ and every interval starts as soon as possible. As a part of his article, Pirlot solves the PIGtoUIG problem, showing when is it possible to transform a PIG model into a UIG model in such a way that the intervals have length $\ell$ for some input $\ell \in \mathbb{N}$. When applied with $\ell = n$, the algorithm ends with success, and the corresponding UIG model is obtained. His algorithm costs $O(n^3)$ time and $O(n^2)$ space.

Some years latter, in 1994, Mitas proposed a solution for the MinSO problem using $O(n)$ time and space [11] (see also [12]). As we shall see in Section 3.3, her algorithm has a flaw and the interval length of the obtained UIG model is not always minimal. Nevertheless, the algorithm correctly solves the PIGtoUIG model in $O(n)$ time and space, yielding a UIG model whose interval length is at most $n$.

In 1995, Corneil et al. [3] devised a simple method for the PIGtoUIG problem that also runs in $O(n)$ time and space. The main goal of [3] was to provide an efficient algorithm for the recognition of PIG graphs. Such algorithm outputs a PIG model that is then transformed into a UIG model whose intervals have length exactly $n$. This last step could have been solved by using Mitas' algorithm. We remark that Mitas' algorithm is harder to prove than the one by Corneil et al., yet the length of the intervals is usually lower than $n$.

In 1999, Bogart and West gave a simple proof of Roberts' PIG=UIG theorem. This proof is quite short and elegant, as it was the main goal pursued by the authors. The proof is constructive and it actually shows how to obtain a UIG model from an input PIG model. However, the extremes of the model so obtained can be of an exponential value in terms of $n$, thus the model occupies $O(n^2)$ space. The construction costs $O(n^2)$

operations, and each operation costs $O(n)$ time in the worst scenario. Thus, $O(n^3)$ time is required. In some sense, the algorithms proposed by Mitas and Corneil et al. also provide us with corresponding not-so-short proofs of Roberts' PIG=UIG theorem.

In 2007, Gardi [6] described an equivalent version of Roberts' PIG=UIG theorem. This theorem also tells us how to obtain a UIG model from an input PIG model. The idea is quite similar to the one by Bogart and West, yet each interval is processed once. Thus, the algorithm requires $O(n)$ operations, but each operation costs $O(n)$ time in the worst case, as the extremes of the intervals could be exponential in terms of $n$. What it is interesting to remark is that Gardi discusses the computational issues of his algorithm. We remark the following statement in [6].

> "the construction given [...] yields a linear-time and space algorithm [...]. This is more efficient than the Bogart–West construction which computes a unit interval model in $O(n^2)$ time [...].
>
> However, these representations are not efficient in the sense that the endpoints of the unit intervals are some arbitrary rationals which may have denominators exponential in $n$. Only Corneil et al. have shown that their breadth-first search recognition algorithm could be used to construct a unit interval model in which each endpoint is rational, with denominator $n$ and numerator lower than $n^2$. Thus, it would be interesting to find a linear-time and space algorithm which computes directly (that is, without the use of breadth-first search) [...]."

As we have seen, the algorithm by Corneil et al. was **not** the only linear time and space algorithm to solve the PIGTOUIG problem whose output model have extremes of length $O(n)$. In fact, the algorithms by Pirlot and Mitas both solve the exact same problem. In particular, Mitas' algorithm does it so in linear time; as she must find some paths in a directed acyclic graph, her algorithm can be classified as one that requires the use of tree traversals. This shows an interesting fact: even though it is well known that interval models are equivalent to interval orders, some researches focus their jobs in graphs and others in orders and, sometimes, there is some ignorance about what has been done in the other community[2].

In 2009, Lin et al. [9] solved Gardi's problem by showing how to transform a PIG model into a UIG model in $O(n)$ time without using any tree-traversal algorithm. Instead, their algorithm does two linear passes through the input PIG model. The UIG model so obtained has length at most $2n$ in the worst case.

Finally, Costa [4] solve a problem related to the MINUIG problem. This problem, called the MINPP, consists of finding the minimum $k$ and $q$ such that $G(\mathcal{I})$ is an induced

[2]Gardi's article was published in a journal that is well known for its excellence and in which every manuscript is subject to at least two strict peer reviews.

subgraph of the $k$-th power of a path with $q$ vertices, for an input PIG model $\mathcal{I}$. As stated by Costa et al., their algorithm finds the minimum feasible value $k^*$ and then, for this fixed $k^*$, it finds takes the minimum feasible value $q$. Note that, in principle, $q$ could be different from the minimum value $q^*$ such that $G(\mathcal{I})$ is an induced subgraph of a power of a path with $q^*$ vertices. However, as proven by Pirlot [13], it matters not the order in which the values $k^*$ and $q^*$ are found, i.e., $q = q^*$. With respect to its complexity, the algorithm by Costa et al. requires $O(n^2)$ time and space.

To end this section we recall that, in few words, our goal is to study how to efficiently solve the MinUIG problem. That is, we must transform the intervals of $\mathcal{I}$ without changing their intersections so that 1. all the intervals have the same length $\ell$, 2. the extremes of all the intervals are integer, and 3. $\ell$ is minimized.

The above description of the MinUIG problem is somehow incomplete if we do not state how the input and output of the problem looks like. A proper interval model $\mathcal{I}$ can be represented by its sequence of extremes. By traversing the real line from $-\infty$ to $\infty$ while writing an `s` for each beginning point and a `t` for each ending point, we obtain a sequence with $2n$ symbols. This sequence is called the *extreme sequence* of $\mathcal{I}$ and somehow identifies $\mathcal{I}$ when the actual values of the extremes are not important. This is the case for our problem, as the goal is to transform $\mathcal{I}$ by moving its extremes. For this reason, we consider the extreme sequence of $\mathcal{I}$ as the actual input of the algorithm. Working directly with the extreme sequence is, however, annoying. To avoid doing so is that we define the equivalence relation above. Note that two PIG models are equivalent if and only if their extreme sequences are equal. As for the output of the problem, we actually do require the positions of the extremes in the UIG model.

Observe that the input extreme sequence is encoded with only $O(n)$ bits while the output UIG model requires $\Omega(n \log n)$ bits, as each extreme must be encoded with $\Omega(\log n)$ bits. This implies that **no** algorithm for the MinUIG and PIGtoUIG problems can run in *linear time*. However, the extreme sequence is usually build from an input graph by invoking the algorithm by Deng et al. [5] that outputs a PIG model requiring $\Theta(n \log n)$ bits. Thus, in the literature it is customary to assume that an input PIG model is encoded with $\Theta(n \log n)$ bits that explains why the term "linear" is used. We continue this tradition for the sake of exposition, i.e., we consider that an algorithm runs in $O(f(n))$ time when it applies $O(f(n))$ operations on integers encoded with $O(\log n)$ bits (see also the discussion in [8]).

## 1.1 Our contributions

The main purpose of this thesis is to solve the MinUIG problem, by fixing Mitas' algorithm. Unfortunately, the so obtained algorithm runs in $O(n^2)$ time. Besides giving a

right solution for the MinUIG problem, we also translate some of the work done by Pirlot and Mitas from the world of semiorders to that of interval graphs, providing alternative proofs that we believe are simpler for those researchers in the graph community. We think that doing this translation is important since, as we have already observed, those researchers who work with either of these approaches sometimes ignore the work done by the other party. This resulted in multiple attempts of solving an already solved "open problem".

We also believe that our algorithm is rather simple both to understand and to implement. In fact, the algorithm applies some well-known subrouting for solving different minimum path problems that are implemented in several libraries and programming languages. So, the work required to obtain an executable implementation is rather small.

Finally, we remark that not only we fix Mitas' algorithm, but we do so by considering the more general problem in which each extreme is required to be separated from the next extreme by an integer threshold. This allows us to solve the MinPP problem easily. All we have to do is to apply our algorithm to the input model with the correct separation parameters.

## 1.2 Review of the previous algorithms

In this section we briefly review some of the algorithms we found in the literature related to the topic of our work.

### 1.2.1 Pirlot's algorithm

Pirlot [13] studies the existence of minimal representations for semiorder. He proves that there exists a UIG model where simultaneously (a) the interval length is minimum among all the equivalent UIG models, (b) each interval starts as soon as possible, and (c) every pair of non-intersecting intervals are separated by a threshold $\varepsilon > 0$. He also shows that all the extremes are integer when $\varepsilon = 1$ which proves the existence of what we call a minimal UIG model. As part of his work, he also shows an algorithm that finds a UIG model of any input semiorder, that we now review.

Pirlot represents a semiorder using a *preference* relation $P$ and an *indifference* relation $I$ on a ground set $A$. Then, $(P, I)$ is a semiorder if and only if there exists a real function $s$ and a nonnegative constant $\ell$ such that

- $aPb$ if and only if $s(a) \geq s(b) + \ell + 1$, and
- $aIb$ if and only if $|s(a) - s(b)| < \ell$

for every $a, b \in A$. In other words, $a$ is "preferred" over $b$ when the interval of $a$ appears completely to the right of the interval of $b$, while $a$ and $b$ are "indifferent" when their corresponding intervals intersect.

To represent $(P, I)$, Pirlot uses a digraph $G$ with vertex set $A$ in which $a \to b$ if and only if either $aPb$ or $aIb$. Note that this digraph has $\Theta(n^2)$ edges; moreover, it contains both arcs $a \to b$ and $b \to a$ when $aIb$ because $I$ is symmetric. Take a cycle $C = a_1 \to \ldots \to a_q \to a_1$ in $G$. This cycle is composed by $k$ edges in $P$ of weight $q+1$ and $j$ edges in $I$ of weight $-q$. By definition, $s(a_i) \geq s(a_{i+1}) + \ell + 1$ for every $a_i P a_{i+1}$, while $|s(a_i) - s(a_{i+1})| < \ell$ for every $a_i I a_{i+1}$. Thus, $s(a_1) \geq s(a_1) + k(\ell+1) - j\ell$ implying that $\ell \geq \frac{k}{j-k}$. As this must happen for every cycle in $G$, it follows that

$$\ell \geq \max \left\{ \frac{|C \cap P|}{|C \cap I| - |C \cap P|} \;\middle|\; C \text{ is cycle of } G \right\}.$$

Note that, by taking $\ell = n$, the above equation is satisfied. The algorithm then takes $n$ as the length of the intervals and finds the position $s(a)$ by solving the maximum path problem with the Bellman-Ford algorithm on $G$, where each edge $a \to b$ has weight either $q+1$ or $-q$ according to whether $aPb$ or $aIb$, respectively.

### 1.2.2 Mitas' algorithm

Mitas' [11] goal was to find the minimal representation of a semiorder with no equal elements. Based on Pirlot's ideas, she represents the semiorder using a digraph that contains a vertex for each interval, while directed edges indicate the separation among the beginning points. However, instead of using a digraph with $\Theta(n^2)$ edges, Mitas uses a trimmed *synthetic graph* of a semiorder. Interestingly enough, the synthetic graph of a semiorder was actually defined by Pirlot in [13] We use the term *trimmed* above because Mitas removes some edges of the synthetic graph so as to remove all its cycles. This allows her to compute the maximum distance from a initial vertex to **all** the remaining vertices in $O(n)$ time. These distances are then taken as the beginning points in the output model. Unfortunately, she removes some essential edges while trimming that in the end prevent her from finding the minimum length (see Section 3.3).

### 1.2.3 Corneil et al. algorithm

Let $\mathcal{I}$ be a PIG model with intervals $I_1, I_2, \ldots, I_n$, where $I_i = (s_i, t_i)$ and $s_1 < s_2 < \ldots < s_n$. The idea of Corneil et al. is to build a special breath-first search (BFS) tree, where the root corresponds to $I_1$. This tree is used, in part, to check if the input graph is actually a PIG graph. However, in this case we already know that the graph is PIG,

thus the construction of the BFS tree can be simplified as follows. For $i = 2, \ldots, n$, let $PREV(I_i)$ be the interval whose ending point appears first from $s_i$. Let $T$ be an ordered tree with vertices $v_1, \ldots, v_n$ such that

- $v_i$ is the parent of $v_j$ if and only if $I_i = PREV(I_j)$ and
- if $v_i$ and $v_j$ are siblings vertices then $v_i$ appears before $v_j$ if and only if $i < j$.

Tree $T$ is a BFS tree of $G$ from the vertex $v_1$ that corresponds to $I_1$. The algorithm by Corneil et al. is really simple. Let $\ell_i$ be the level of $v_i$ in $T$, $k_i$ be the position of $v_i$ in a postorder traversal of $T$, and define $a_i = n\ell_i + k_i$ and $b_i = n(\ell_i + 1) + k_i$. Then, the model $\{(a_i, b_i) \mid 1 \leq i \leq n\}$ is a UIG model equivalent to $\mathcal{I}$ [3].

It is not hard to compute all the values of $PREV$ in $O(n)$ time. Therefore, Corneil et al. algorithm runs in $O(n)$ time and space.

### 1.2.4 The UIG=PIG proof by Bogart and West

In their work, Bogart and West [1] show a simple constructive proof of the PIG=UIG theorem. Suppose a PIG model $\{(s_i, t_i) \mid 1 \leq i \leq n\}$ is given, where $s_1 < \ldots < s_n$. At the $i$-th step of the algorithm, some section of the real line is stretched or shrunk in such a way that the first $i$ intervals have length 1. For $i = 1$, stretch or shrink the real line in $[s_1, t_1]$ so that $t_1 - s_1 = 1$. Note that this operation can affect not only $(s_1, t_1)$ but any other interval that intersects it. For step $i+1$, shrink or stretch $[t_i, t_{i+1}]$ so that $s_{i+1} - t_{i+1} = 1$. The main observation is that this operation is always possible without affecting the length of the previous intervals. At the end, a UIG model is obtained.

### 1.2.5 Gardi's UIG=PIG proof

Gardi's [6] idea is similar to that of Bogart and West, but he stretches or shrinks entire maximal cliques at each step $i$, as follows. Again, suppose a PIG model $\{(s_i, t_i) \mid 1 \leq i \leq n\}$ is given, where $s_1 < \ldots < s_n$. For $i = 1$, take the maximal clique including the interval $(s_1, t_1)$ and make each of its interval of length 1. This is obviously possible without altering the order of the extremes. Then, for $i+1$, take the maximal clique $C_{i+1}$ that includes the first interval $(s_j, t_j)$ not contained in $C = C_1 \cup \ldots \cup C_i$. Is not hard to see that we can make the intervals in $C_{i+1} \setminus C$ of length 1 without affecting the order of the extremes.

### 1.2.6 The algorithm by Lin et al.

The algorithm by Lin et al. [9] works by "separating" the extremes of the input interval model $\mathcal{I}$ by means of successive insertion of new intervals, until the obtained model $\mathcal{I}^*$ is that of a *power of path*. That is, $\mathcal{I}^*$ represents the $k$-th power of the path with $q$ vertices, denoted by $P_q^k$. It is not hard to see that $\mathcal{U} = \{(2i, 2(i+k)+1) \mid 1 \leq i \leq q\}$ is a UIG model representing $P_q^k$, i.e., $\mathcal{U}$ is equivalent to $\mathcal{I}^*$. The main property of the separation is that $\mathcal{I}$ still appears *within* $\mathcal{I}^*$. So, if we remove the intervals of $\mathcal{I}^*$ added during the separation from the corresponding unitary model $\mathcal{U}$, we recover $\mathcal{I}$ but with a unitary representation.

The main observation for the separation algorithm is that a extreme sequence represents $P_q^k$ if and only if it has the pattern $\mathbf{s}_1 \ldots \mathbf{s}_{k+1}\mathbf{t}_1\mathbf{s}_{k+2}\mathbf{t}_2 \ldots \mathbf{s}_q\mathbf{t}_{q-k} \ldots \mathbf{t}_q$ (the subindices are written for the sake of exposition only). So, the separation algorithm works by inserting an ending point between consecutive beginning points in the *middle* and then inserting beginning points between consecutive ending points in the middle in such a way that, when the separation concludes, each beginning point in the middle is followed by an ending point.

To describe the separation algorithm in more precise terms, suppose $\mathcal{I} = \{(s_i, t_i) \mid 1 \leq i \leq n\}$ is given, where $s_1 < \ldots < s_n$. The extremes in $[t_1, s_n]$ are said to be *middle* extremes. Traverse the extremes from left to right, and while there are not separated middle ending points do:

1. Let $t_i$ be the leftmost non-separated middle ending point and $(s_j, t_j)$ be the interval such that $s_j$ precedes $t_i$ in $\mathcal{I}$.
2. Insert in $\mathcal{I}$ a new interval $(s, t)$, placing $s$ immediately after $t_i$ and $t$ immediately after $t_j$.

In the model so obtained there are no more pairs of middle consecutive ending points. Now we repeat the procedure but this time for the beginning points while traversing from right to left.

As described in this section, the algorithm takes $O(n^2)$ time, as it must insert all the separating intervals. The actual implementation works in a virtual manner, by taking account of the number of intervals that should be inserted, but it inserts none of them. This is the reason why the algorithm costs $O(n)$ time and $O(n)$ space.

# 2 Preliminaries

A *graph* $G$ is an ordered pair $(V(G), E(G))$ where $V(G)$ is a finite and nonempty set and $E(G)$ is a set of unordered pairs of different vertices. The elements of $V(G)$ are the *vertices* of $G$ while the elements or $E(G)$ are its *edges*; $V(G)$ and $E(G)$ are the *vertex set* and *edge set* of $G$, respectively. The idea is to represent a symmetric relationship where each vertex represents an abstract object and each edge represents a connection between two objects. We write either $vw$ or $(v, w)$ to denote the edge *between* the vertices $v$ and $w$, regardless of whether it belongs to $E$ or not. When $vw \in E$, we say that $v$ and $w$ are *adjacent*, or that $v$ is *adjacent to* $w$, or that $v$ is a *neighbor* of $w$. Note that, for us, $v_i$ cannot be adjacent to itself (i.e., $G$ has no loops), while the edge $v_iv_j$ is the same as the edge $v_jv_i$ (i.e., $G$ is undirected).

Let $G$ be a graph. Any graph $(V, E)$ such that $V \subseteq V(G)$ and $E \subseteq E(G)$ is a *subgraph* of $G$. When $E$ contains all the edges of $G$ between the vertices in $V$, then $(V, E)$ is an *induced subgraph* of $G$. In such case, we also say that $(V, E)$ is the subgraph of $G$ *induced* by $V$, and we write $G[V]$ to denote this graph. For $V \subset V(G)$, we write $G \setminus V$ to denote the graph $G[V(G) \setminus V]$, i.e., the graph that is obtained by removing all the vertices in $V$.

A *walk* $P$ in a graph $G$ is a sequence of vertices $v_1, \ldots, v_p$ such that $v_i$ is adjacent to $v_{i+1}$ for every $1 \leq i < p$. We say that $P$ *goes from* $v_1$ *to* $v_p$, $v_1$ and $v_p$ being the *first* and *last* vertices of $P$, respectively. For the sake of simplicity, we also consider each path a sequence of edges, where each edge $v_iv_{i+1}$ is an *edge of* $P$. The *length* of $P$ is $p-1$, i.e., the number of edges it contains. We say that $P$ is a *closed walk* when $v_1 = v_p$, that $P$ is a *path* when $v_i \neq v_j$ for every $1 \leq i < j \leq p$, and that $P$ is a *cycle* when it is a closed walk and $v_1, \ldots, v_{p-1}$ is a path. A graph $G$ is *connected* when there is a path that goes from $v$ to $w$, for every $v, w \in V(G)$.

In this thesis we also work with directed graphs. A *digraph* $D$ is a pair $(V(D), E(D))$ where $V(D)$ is a finite and nonempty set and $E(D)$ is a set of ordered pairs of different vertices. As with graphs, $V(D)$ and $E(D)$ are respectively the vertex and edge sets of $D$, while their elements are the *vertices* and *edges* of $D$. Note that in this case, a digraph represents a non-symmetric relationship. We write $(v, w)$ to denote the edge that *goes from* $v$ *to* $w$, regardless of whether it belongs to $E(D)$ or not. When we know that $(v, w) \in E(D)$, we simply write $v \to w$ to indicate this fact, while we write $v \not\to w$ when $(v, w) \notin V(D)$. As with graphs, $v$ and $w$ are said to be *adjacent*, while $v$ is the

*in-neighbor* of $w$ and $w$ is the *out-neighbor* of $v$. Again, for us, digraphs cannot have loops, i.e., $v \not\to v$ for $v \in V(D)$.

A *walk* $P$ in a directed graph $D$ is a sequence of vertices $v_1, \ldots, v_p$ such that $v_i \to v_{i+1}$ for every $1 \leq i < p$. Path $P$ *goes from* $v_1$ *to* $v_p$, $v_1$ and $v_p$ being the *first* and *last* vertices of $P$, respectively. For the sake of simplicity, we consider each path also a sequence of edges, where each edge $v_i \to v_{i+1}$ is an *edge of* $P$. We extend the terminology of paths from graphs to digraphs, as follows. The *length* of $P$ is $p-1$, i.e., the number of edges it contains. When $v_1 = v_p$, we say that $P$ is a *closed walk*. A *path* is a walk that contains no repeated vertices, while a *cycle* is a closed walk whose first vertex is the unique repeated vertex.

The *underlying graph* $G(D)$ of a digraph $D$ is the graph that is obtained by replacing each edge of $D$ with a non-oriented edge, i.e., the edge set of $D$ is replaced by a set of unordered pairs. A digraph $D$ is *strongly connected* when there is a path that goes from $v$ to $w$, for every $v, w \in V(G)$, while it is *connected* when its underlying graph is connected. Clearly, every strongly connected digraph is connected.

A *weighing* of digraph $D$ is a function $f\colon E(D) \to \mathbb{R}$ that assigns a *weight* $f(e)$ to each edge $e \in E(D)$. For any walk $P$, we write $f(P) = \sum_{v_i \to v_j \in P} f(v_i \to v_j)$, and we say that $f(P)$ is the *$f$-weight* of $P$. The *$f$-distance* between two vertices $v$ and $w$, denoted by $d_f(v, w)$, is the maximum among the $f$-weights of the walks that go from $v$ to $w$. Note that the $f$-distance is not well defined when either there is no walk from $v$ to $w$ or the $f$-weights of the walks between $v$ and $w$ are unbounded, in such case, $d_f(v, w) = \infty$. To avoid having unbounded distances, we define the *path $f$-distance* from $v$ to $w$, denoted by $d^*_f(v, w)$, to be the maximum among the weights of the paths that go from $v$ to $w$. It is well known that $d_f(v, w) < \infty$ for every $v, w$ if and only if no cycle of $D$ has positive weight [2]. In such case, $d_f = d^*_f$. For the sake of notation, we drop the subscript $f$ from $d_f$ and $d^*_f$ when $f$ is clear from context.

An *interval (IG) model* is a set $\mathcal{I} = \{I_1, \ldots I_n\}$ of open intervals in $\mathbb{R}$. An *interval model* is a *proper interval (PIG) model* when no interval properly contains another, while it is a *unit interval (UIG) model* when all the intervals have the same length. We write $I_i = (s_i, t_i)$, where $s_i$ and $t_i$ are the *extremes* of $I_i$; $s_i$ and $t_i$ are the *beginning* and *ending points* of $I_i$, respectively. The *extremes* of $\mathcal{I}$ is the set of all the extremes of the intervals in $\mathcal{I}$. As it is common practice, we assume all the extremes of $\mathcal{I}$ are different, as it does not affect the interval graph defined by $\mathcal{I}$ (cf. below). Two *extremes* $e_1e_2$ of $\mathcal{I}$ are said to be *consecutive* when $e_2$ appears immediately after $e_1$ in a left to right traversal of $\mathcal{I}$; note that the order is important, $e_2e_1$ is not consecutive when $e_1e_2$ is consecutive. The *reverse* of $\mathcal{I}$ is $\{(-t_i, -s_i) \mid (s_i, t_i) \in \mathcal{I}\}$; that is, the reverse of $\mathcal{I}$ is the model that we obtain by mirroring all the intervals. Traversing $\mathcal{I}$ from left to right and placing an **s** for each beginning point and a **t** for each ending point we obtain the *extreme sequence*

of $\mathcal{I}$. Clearly, two non-equal models can have the same extreme sequence, in such case we say these models are *equivalent*.

For any interval model $\mathcal{I}$, the *interval (IG) graph* of $\mathcal{I}$ is the graph $G(\mathcal{I})$ that has one vertex $v_i$ for each $1 \leq i \leq n$ such that $v_i v_j \in E(G)$ if and only if $I_i \cap I_j \neq \emptyset$ for every $1 \leq i < j \leq n$. When $\mathcal{I}$ is a proper (resp. unit) interval model, the graph $G(\mathcal{I})$ is a *proper (resp. unit) interval (PIG, resp. UIG) graph*. We say that $\mathcal{I}$ *represents* or is a *model* of $G(\mathcal{I})$, while $G(\mathcal{I})$ *admits* $\mathcal{I}$. It is not hard to see that the reverse of $\mathcal{I}$ also represents $G(\mathcal{I})$. Similarly, every two equivalent models represent the same interval graph. It is well known that every PIG model admits exactly two PIG models up to equivalence, one the reverse of the other [14]. Also, Roberts [14] proved that every PIG model is equivalent to some UIG model; this property is known as the *PIG=UIG theorem*.

**Theorem 1** ([14])**.** Every PIG model is equivalent to some UIG model.

We say that $\ell$ is the *length* of a UIG model $\mathcal{I}$ to mean that all the intervals of $\mathcal{I}$ have length $\ell$. Pirlot [13] defined a UIG model $\mathcal{I} = \{(s_i, s_i + \ell) \mid 1 \leq i \leq n\}$ as *minimal* when

- $\ell \leq \ell'$, and
- $s_i \leq s'_i$

for every equivalent UIG model $\mathcal{I}' = \{(s'_i, s'_i + \ell') \mid 1 \leq i \leq n\}$. The main theorem in [13] strengthens the PIG=UIG theorem by stating that every UIG model is equivalent to a minimal UIG model.

**Theorem 2** ([13])**.** Every UIG model is equivalent to a minimal UIG model.

The *k-th power* of a graph $G$ is the graph $G^k = (V(G), E^k)$ where $E^k$ is the set of edges $v_i v_j$ such that there is a path from $v_i$ to $v_j$ of length at most $k$ in $G$. The *path graph* $P_q$ is the graph with $q$ vertices $v_1, \ldots, v_q$ where $v_i$ is adjacent to $v_{i+1}$ for every $1 \leq i < q$. The *k-th power of path* $P_q^k$ is simply the $k$-th power of the path graph $P_q$. Every UIG graph is an induced subgraph of a power of a path, and vice versa [10].

**Theorem 3** ([10])**.** A graph is a UIG graph if and only if it is an induced subgraph of a power of a path.

In an analogy to Pirlot's definition, we say that $P_q^k$ is a *minimal extension* of $\mathcal{I}$ when $G(\mathcal{I})$ is an induced subgraph of $P_q^k$ and:

- $k \leq j$, and
- $q \leq r$

for every $P_r^j$ such that $G(\mathcal{I})$ is an induced subgraph of $P_r^j$.

# 3 Minimal representations of unit interval graphs

In this chapter we study the problem of finding a minimal UIG model equivalent to an input PIG model. We begin with Section 3.1, where we define what the length constraints of a UIG model are, that allow us to generalize the synthetic representation given by Pirlot [13]. We show necessary and sufficient conditions for a PIG model $\mathcal{I}$ to be equivalent to a UIG model of length $\ell$ satisfying such constraints, when $\ell$ is given. As a consequence, we obtain an $O(n^2)$ time algorithm for finding such an UIG model when $\mathcal{I}$ and $\ell$ are given as input.

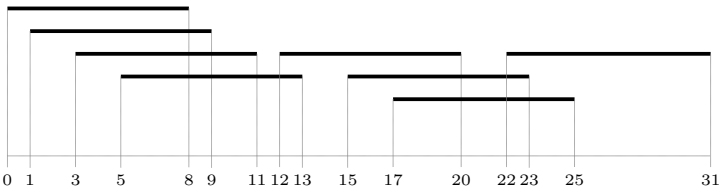
In Section 3.2 we devise an $O(n^2)$ time algorithm that actually finds the minimum possible value of $\ell$ such that $\mathcal{I}$ is equivalent to a UIG model of length $\ell$ that satisfies the required separations constraints. Combining this algorithm with the one in Section 3.1, an $O(n^2)$ time algorithm that outputs a minimal UIG model equivalent to $\mathcal{I}$ is obtained. In Section 3.3 we discuss the algorithm given by Mitas [11], and we show that, although the algorithm finds a UIG model equivalent to $\mathcal{I}$, such a model is not necessarily minimal as claimed. Finally, in Section 3.4 we apply our algorithm in order to find a minimal extension of $\mathcal{I}$. We believe our algorithm is conceptually simpler than the one in [4] that also runs in $O(n^2)$ time.

## 3.1 Finding a representation with constraints

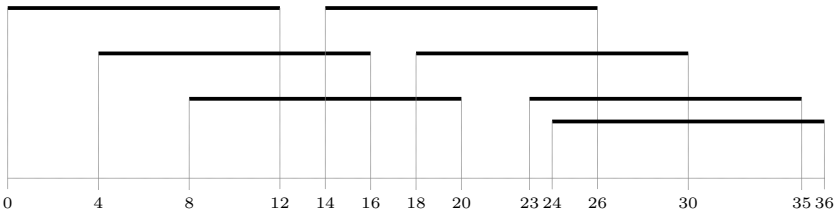
In this section we study the problem of finding a UIG model $\mathcal{I}$ of a PIG graph that has a given length $\ell$ and satisfies certain separation constraints. To describe the length and separation constraints, we use a quadruple $L = (\ell, \nu, \eta, \sigma)$, where $\ell$ is a natural number and $\eta$, $\nu$, and $\sigma$ are functions in $\{1, \ldots, n\} \to \mathbb{N}$. We say that an interval model $\mathcal{I} = \{(s_i, t_i) \mid 1 \leq i \leq n\}$ is an *L-UIG* model when $\mathcal{I}$ is a UIG model of length $\ell$ such that:

- $s_j \geq t_i + \nu(i)$ whenever $t_i s_j$ are consecutive in $\mathcal{I}$,
- $t_j \geq s_i + \eta(i)$ whenever $s_i t_j$ are consecutive in $\mathcal{I}$, and
- $s_{i+1} \geq s_i + \sigma(i)$ for every $1 \leq i < n$.

The quadruple $L$ is referred to as the *length constraints* of $\mathcal{I}$. Figure 3.1 shows $L$-UIG models for different length constraints $L$. Using our new terminology, the purpose of this section is to find an $L$-UIG model equivalent to a PIG model $\mathcal{I}$, when both $\mathcal{I}$ and $L$ are given as input.




**(a)** An $(\ell, \nu, \eta, \sigma)$-UIG model for $\ell = 8$, $\nu = 1$, $\eta = 1$, and $\sigma(1) = 1$, $\sigma(2) = 2$, $\sigma(3) = 1$, $\sigma(4) = 4$, $\sigma(5) = 2$, $\sigma(6) = 2$, $\sigma_1(7) = 4$.




**(b)** An $(\ell, \nu, \eta, \sigma)$-UIG model for $\ell = 12$, $\nu = 2$, $\eta = 2$ and $\sigma = 1$.

**Figure 3.1** – *Two models satisfying different $L$-constraints. Intuitively, $\ell$ indicates the length of the intervals; $\nu(i)$ is the minimum space that must follow $t_i$ before the next extreme $s_j$ can appear; $\eta(i)$ is the minimum space that must follow $s_i$ before the next extreme $t_j$ can appear; and $\sigma(i)$ is the minimum separation between $s_i$ (or $t_i$) and the next beginning (or ending) point $s_{i+1}$ (or $t_{i+1}$).*

The main tool of this section is a weighing function of a variation of the synthetic graph defined by Pirlot [13]. The *synthetic graph* of $\mathcal{I}$, or simply the *synthesis* of $\mathcal{I}$, is the digraph $S(\mathcal{I})$ with vertex set $\{v_i \mid 1 \leq i \leq n\}$ such that:

(i) $v_i \to v_j$ if and only if $t_i s_j$ are consecutive in $\mathcal{I}$

(ii) $v_i \to v_j$ if and only if $s_i t_j$ are consecutive in $\mathcal{I}$

(iii) $v_i \to v_{i+1}$ if and only if $1 \leq i < n$.

Figure 3.2 shows an example of the synthesis for the PIG model in Figure 3.1(a). Following Pirlot, we refer to the edges of type (i) and (ii) respectively as *noses* and *hollows*, while type (iii) edges are referred to as *steps*. For the sake of notation, we omit the parameter $\mathcal{I}$ from $S(\mathcal{I})$ when no ambiguities arise. It should be noted that every pair of equivalent models have isomorphic synthetic graphs.

For a length constraints $L$, the *$L$-weighing* of $S$ is the function $L$ such that:

- $L(v_i \to v_j) = \ell + \nu(i)$ for every nose $v_i \to v_j$
- $L(v_i \to v_j) = -\ell + \eta(i)$ for every hollow $v_i \to v_j$, and
- $L(v_i \to v_{i+1}) = \sigma(i)$ for every step $v_i \to v_{i+1}$.

The overloaded notation for $L$ is intentional, and it indicates how the constraints $L$ impose a weighing $L$. Figure 3.2 shows the $L$-weighing of $S$.

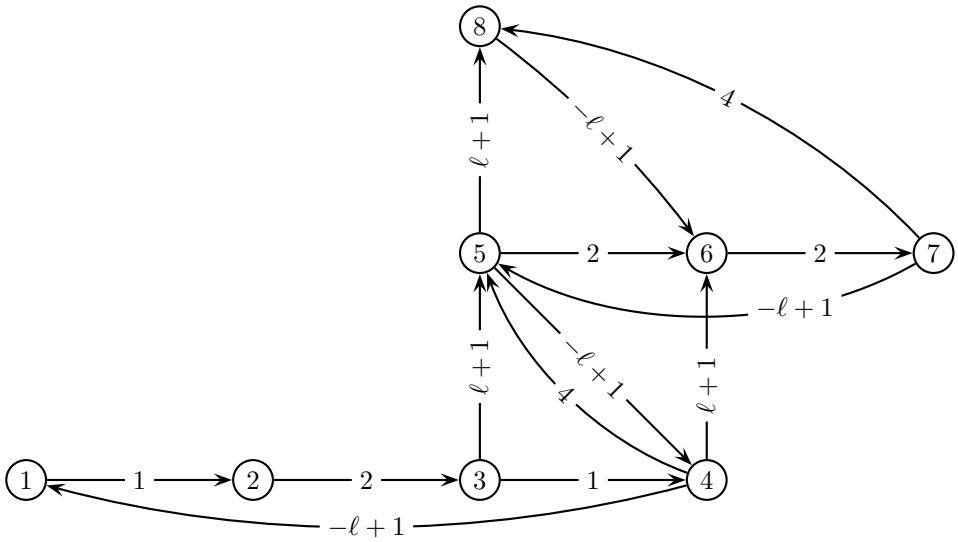



**Figure 3.2** – *The synthesis of the model in Figure 3.1(a). The value inside each edge $v_i \to v_j$ corresponds to $L(v_i \to v_j)$ for the length constraints $L$ given in Figure 3.1(a).*

The $L$-weighing of $S$ has an intuitive interpretation. Suppose $t_i s_j$ are consecutive in an $L$-UIG model $\mathcal{I} = \{(s_i, t_i) \mid 1 \leq i \leq n\}$. By definition, $s_j \geq t_i + \nu(i)$, thus $s_j$ appears at distance at least $\ell + \nu(i)$ from $s_i$. In $S$ there is a nose from $v_i \to v_j$ and, by definition, $L(v_i \to v_j)$ is precisely $\ell + \nu(i)$. Similar conditions apply to hollows and steps. So, in other words, the $L$-weighing correctly models the minimum separation between the beginning points of any $L$-UIG model equivalent to $\mathcal{I}$.

The main theorem of this section states that $\mathcal{I}$ is equivalent to an $L$-UIG model if and only if $\ell$ is large enough so as to fulfill all the restrictions imposed by $\nu$, $\eta$ and $\sigma$. One such $L$-model can be obtained by looking at the $L$-distances in $S$. With that purpose in mind, we define a special UIG model for each $S$ and $L$; the UIG model $\mathcal{U}_L(\mathcal{I}) = \{(s_i^u, t_i^u) \mid 1 \leq i \leq n\}$ is such that:

- $s_i^u = d_L(v_1, v_i)$
- $t_i^u = s_i + \ell$.

when $d_L = d_L^*$, and it is undefined otherwise. Note that $d_L^*(v_1, v_i)$ is well defined for every $1 \leq i \leq n$ because the path $v_1, v_2, \ldots, v_i$ exists. Figure 3.3 shows $\mathcal{U}_{(10,\nu_1,\eta_1,\sigma_1)}$ for the model in Figure 3.1(a). The following theorem shows that $\mathcal{I}$ is equivalent to an $L$-UIG model if and only if $\mathcal{U}_L$ is well defined.

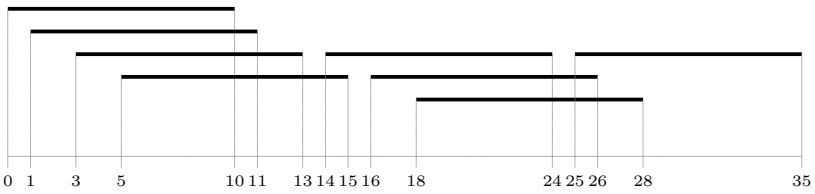



**FIGURE 3.3** – $\mathcal{U}_L(\mathcal{I})$ *for the model in Figure 3.1(a) with* $L = (10, \nu, \eta, \sigma)$*, where* $\nu$*,* $\eta$*, and* $\sigma$ *are as in Figure 3.1(a).*

**Theorem 4.** Let $\mathcal{I}$ be a PIG model and $L$ be some length constraints of it. Then, the following statements are equivalent.

(i) $\mathcal{I}$ is equivalent to an $L$-UIG model.

(ii) $S$ contains no cycles of positive $L$-weight.

(iii) $\mathcal{U}_L(\mathcal{I})$ is equivalent to $\mathcal{I}$.

*Proof.* Let $\mathcal{I} = \{(s_i, t_i) \mid 1 \leq i \leq n\}$ and consider each implication.

(i) $\Rightarrow$ (ii). Let $C = w_1, \ldots, w_k, w_1 = w_{k+1}$ be a cycle of maximum $L$-weight in $S$ and $(s(w_i), t(w_i))$ be the interval represented by $w_i$ in an $L$-UIG model equivalent ot $\mathcal{I}$. Recall that, by definition, $s(w_{i+1}) \geq s(w_i) + L(w_i \to w_{i+1})$ for every $1 \leq i \leq k$ as explained above. Then, by induction, $s(w_1) \geq s(w_1) + \sum_{i=1}^{k} L(w_i \to w_{i+1}) = s(w_1) + L(C)$.

(ii) $\Rightarrow$ (iii). By hypothesis, $S$ has no cycles of positive length, thus $d_L = d_L^*$ and, so, $\mathcal{U}_L(\mathcal{I})$ is well defined. Call $s_i^u$, $t_i^u$ to the extremes of $\mathcal{U} = \mathcal{U}_L(\mathcal{I})$ as in its definition. Note that it suffices to show that any two consecutive extremes of $\mathcal{I}$ appear in the same order in $\mathcal{U}$. Then, only three cases are required:

**Case 1:** $s_i, s_{i+1}$ or $t_i, t_{i+1}$ are consecutive in $\mathcal{I}$. Since $v_i \to v_{i+1}$ is a step of $S$ with $L(v_i \to v_{i+1}) = \sigma(i) > 0$, it follows that $s_i^u = d_L(v_1, v_i) < d_L(v_1, v_{i+1}) = s_{i+1}^u$.

**Case 2:** $t_i s_j$ are consecutive in $\mathcal{I}$. Then, $v_i \to v_j$ is a nose of $S$ with $L(v_i \to v_j) = \ell + \nu(i)$, thus $s_j^u = d_L(v_1, v_j) \geq d_L(v_1, v_i) + \ell + \nu(i) = s_i^u + \ell + \nu(i) = t_i^u + \nu(i) > t_i^u$.

**Case 3:** $s_i t_j$ are consecutive in $\mathcal{I}$. Then, $v_i \to v_j$ is a hollow of $S$ with $L(v_i \to v_j) = -\ell + \eta(i)$, thus $t_j^u = s_j^u + \ell = d_L(v_1, v_j) + \ell \geq d_L(v_1, v_i) + \eta(i) = s_i^u + \eta(i) > s_i^u$.

(iii) $\Rightarrow$ (i). We show that $\mathcal{U}_L(\mathcal{I})$ is an $L$-UIG model. Note that, since $\mathcal{U}_L(\mathcal{I})$ is equivalent to $\mathcal{I}$, then it must be well defined.

**Case 1**: If $s_i s_{i+1}$ are consecutive in $\mathcal{I}$ then $d_L(v_1, v_{i+1}) \geq d_L(v_1, v_i) + \sigma(i)$, hence $s^u_{i+1} \geq s^u_i + \sigma(i)$ when $s^u_i, s^u_{i+1}$ are consecutive in $\mathcal{U}_L(\mathcal{I})$.

**Case 2**: If $t_i s_j$ are consecutive in $\mathcal{I}$ then $d_L(v_1, v_j) \geq d_L(v_1, v_i) + \ell + \nu(i)$, hence $s^u_j \geq s^u_i + \ell + \nu(i) = t^u_i + \nu(i)$ when $t^u_i s^u_j$ are consecutive in $\mathcal{U}_L$.

**Case 3**: If $s_i t_j$ are consecutive in $\mathcal{I}$ then $d_L(v_1, v_j) \geq d_L(v_1, v_i) - \ell + \eta(i)$, hence $t^u_j = s^u_j + \ell \geq s^u_i + \eta(i)$ when $s^u_i t^u_j$ consecutive in $\mathcal{U}_L$.

And this is the definition of an $L$-UIG model. □

Theorem 4 yields an $O(n^2)$ time algorithm to determine if $\mathcal{I}$ is equivalent to an $L$-UIG model. Just build the synthesis $S$ weighed with $L$ and apply the Bellman-Ford algorithm on it so as to find $d^*_L(v_1, v_i)$ for every $1 \leq i \leq n$. If Bellman-Ford fails, then $S$ has a cycle of positive $L$-weight and, hence, $\mathcal{I}$ has no equivalent model satisfying the constraints. Otherwise, we can build $\mathcal{U}_L(\mathcal{I})$ in $O(n)$ time by following the definition. Since $S$ has only $O(n)$ edges, the total time required is $O(n^2)$.

## 3.2 Finding the minimum interval length

In this section we are interested in finding the minimum value $\ell_*$ such that $\mathcal{I}$ is equivalent to an $(\ell_*, \nu, \eta, \sigma)$-UIG model, when $\nu$, $\eta$ and $\sigma$ are given as input. In other words, by Theorem 4, we ought to find the minimum $\ell_*$ such that $S(\mathcal{I})$ has no cycles with positive $L_*$-weight, for $L_* = (\ell_*, \nu, \eta, \sigma)$. Note that, by Theorem 4, $\mathcal{U}_{L_*}(\mathcal{I})$ would be a model equivalent to $\mathcal{I}$, while $\mathcal{I}$ is equivalent to no $(\ell, \nu, \eta, \sigma)$-UIG model for $\ell < \ell_*$.

A rough idea on how to find $\ell_*$ is as follows. Let $L = (\ell, \nu, \eta, \sigma)$. Clearly, the $L$-weight of any cycle in $S$ depends on the four parameters $\ell$, $\nu$, $\eta$, $\sigma$. In this section $\nu$, $\eta$, and $\sigma$ are given as the input of our algorithm, while $\ell$ is an indeterminate whose optimum value we should compute. So, we find convenient to think of the $L$-weight of a given cycle as function on $\ell$. Of course, to guarantee that all the cycles of $S$ have a non-positive $L$-weight, it suffices to show that the $L$-weight of one cycle $C$ of maximum $L$-weight is at most 0. So, we can write an equation asking the $L$-weight of $C$ to be 0; then by solving this equation with $\ell$ as its unique indeterminate value, we obtain $\ell_*$

To analyze how the $L$-weight of a cycle look like, we use the pictorial representation of $\mathcal{I}$ in which all the intervals are drawn according to their row (or height). This idea was first used by Mitas [11]. Let $\mathcal{I} = \{(s_i, t_i) \mid 1 \leq i \leq n\}$ with $s_1 < \ldots < s_n$, and

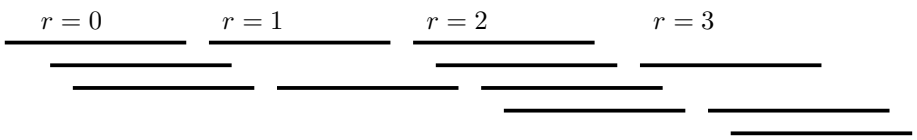



**Figure 3.4** – *Row values for a UIG model.*

recall that $S$ has a vertex $v_i$ representing $(s_i, t_i)$. The *row* of $v_i$ is given by the function $r\colon V(S) \to \mathbb{N}$ such that:

$$r(v_i) = \begin{cases} 0 & \text{if } t_1 > s_i \\ 1 + r(v_j) & \text{otherwise} \end{cases}$$

where $t_j$ is the rightmost ending point before $s_i$. Figure 3.4 shows an example of the row values in a UIG model. Note that, as the picture shows, the intervals appear *in columns* according to their corresponding values of $r$. Hence, $r$ gives as an idea of how far to the right the intervals appear from the leftmost interval. We can compute $r$ in $O(n)$ time with a left to right traversal as follows: start setting $r = 0$ until $t_1$ is found, then assign $r = 1$ until $t_i$ is found for $t_1 s_i$ consecutive ...and so on.

The reason why $r$ is called the *row* function has to do with an idea of Mitas [11], who organizes the vertices in a matricial picture (see Figure 3.8). In this picture, all the vertices with the same $r$ value appear in the same *row* of the matrix, while the vertices in the same row are drawn from left to right as they appear in the model. (Mitas uses a trimmed synthetic graph, and she imposes some other conditions in the picture; see Section 3.3.)

It is not hard to see by definition that if $v_i \to v_j$ is a nose, then $r(v_j) = r(v_i) + 1$. In other words, any time a nose is traversed in any walk $P$ of $S$, the value of $r$ is *increased* by 1. On the other hand, if $v_j \to v_i$ is a hollow, then either $r(v_i) = r(v_j) - 1$ or $r(v_i) = r(v_j)$. That is, each hollow either *decreases* by 1 or leaves *constant* the value of $r$ when is traversed in $P$. Finally, if $v_i \to v_j$ is a step, then either $r(v_j) = r(v_i) + 1$ or $r(v_i) = r(v_j)$. So steps either *increase* by 1 or leave *constant* the value of $r$ for $P$. The hollows that leave constant the value of $r$ are said to be $\pi$-*hollows*, while the steps that increase the value of $r$ are said to be $\pi$-*steps*. In general, we refer to $\pi$-steps and $\pi$-hollows as $\pi$-*edges*.

Recall that our goal is to find those cycles of $S$ with maximum $L$-weight. The row function provides us with some important information about the cycles of $S$; when $S$ is traversed one vertex at a time, the row of the next vertex either remains the same, or increases by one, or decreases by one. Since a cycle is a walk from a vertex to itself, the number of times the row is increased must equal the number of times the row is decreased. Thus, immediately, we see that the number of hollows must be at least the

number of noses plus $\pi$-steps. To see how much more hollows a cycle can have, we study how many $\pi$-edges does a cycle have. The next lemma shows that every cycle must have at least one $\pi$-edge; here $S^-(\mathcal{I})$ is obtained from $S(\mathcal{I})$ by removing all its $\pi$-edges. As usual, we do not write the parameter $\mathcal{I}$ of $S^-$ when it is clear from context.

**Lemma 5** (see also [11])**.** For every PIG model $\mathcal{I}$, there are no cycles in $S^-(\mathcal{I})$.

*Proof.* Let $P = w_1, \ldots, w_k$ be a path in $S^-$ such that $w_k$ is the only vertex in $P$ with $r(w_1) = r(w_k) = r_0$. Each vertex $w_i$ corresponds to some vertex $v_{f(i)}$ in $S^-$, for a function $f$; we prove by induction on $k$ that $f(1) \leq f(k)$, where equality holds if and only if $k = 1$. The base case $k = 1$ is trivial. For the base case $k = 2$, in which $P$ has just one edge, note that such an edge must leave $r$ constant. In $S^-$ there are no $\pi$-hollows, thus $w_1 \to w_2$ is a step and, by definition, $f(2) = f(1)+1$. For the inductive case $k > 2$, recall that $|r(w_i) - r(w_{i+1})| \leq 1$ for every $1 \leq i < k$. Hence $r(w_2) = r(w_{k-1}) = r_0 \pm 1$ and only two cases need to be analyzed.

**Case 1:** $w_1 \to w_2$ is a nose and $w_{k-1} \to w_k$ is a hollow. As there is a path from $w_2$ to $w_{k-1}$, we know by inductive hypothesis that $f(2) \leq f(k-1)$. This means that $s_{f(2)} \leq s_{f(k-1)}$ where $s_i$ is the beginning point of the interval of $\mathcal{I}$ that corresponds to $v_i$ $(1 \leq i \leq n)$. By definition we also know that $t_{f(1)} < s_{f(2)}$ and $s_{f(k-1)} < t_{f(k)}$, hence $f(1) < f(k)$.

**Case 2:** $w_1 \to w_2$ is a hollow and $w_{k-1} \to w_k$ is a nose. Again, by inductive hypothesis, $f(2) \leq f(k-1)$ and $t_{f(2)} < t_{f(k-1)}$ while, by definition, $s_{f(1)} < t_{f(2)}$ and $t_{f(k-1)} < s_{f(k)}$.

Note that any closed walk $W$ from $v_i$ to $v_j$ can be decomposed into paths $P_1, \ldots, P_j$ such that the first and last vertices of each $P_i$ are the only vertices with row $r_0$, for some $r_0$. Then, by induction, $i < j$. □

By the lemma above, every cycle of $S$ has at least one $\pi$-edge. In the following we prove that, in fact, every cycle contains **exactly** one $\pi$-edge. Consequently, we can state that every cycle contains exactly one more hollow than noses (cf. below). For this we need a second function $c\colon V(S^-) \to \mathbb{N}$, called the *column function*, such that $c(v_1) = 0$ and

$$c(v_i) = \max \left\{ c(v_j), c(v_k) + 1, c(v_{i-1}) + 1 \;\middle|\; \begin{array}{l} v_j \to v_i \text{ is a nose of } S^-, \\ v_k \to v_i \text{ is a hollow of } S^-, \text{ and} \\ v_{i-1} \to v_i \text{ is a step of } S^-. \end{array} \right\}.$$

A nose $v_j \to v_i$ (resp. hollow $v_k \to v_i$, step $v_{i-1} \to v_i$) needs not exist in $S^-$; in such a case, $c(v_j)$ (resp. $c(v_k) + 1$, $c(v_{i-1}) + 1$) is not taken into account in the above definition of $c(v_i)$. Note that $c$ is well defined because $S^-$ has no cycles by Lemma 5; thus, $c$ can be easily computed with the algorithmic technique of dynamic programming. Figure 3.5

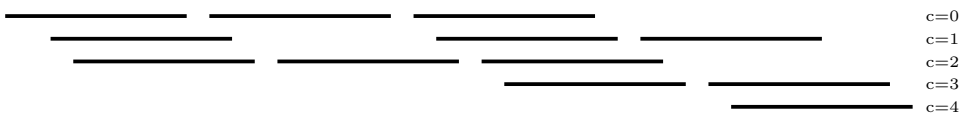



**Figure 3.5** – *Column values for the UIG model of Figure 3.4*

shows the values of $c$ for the model in Figure 3.4. Even though Mitas does not use the term *column*, her pictorial description of $\mathcal{I}$ consists of putting each $v_i$ in a matrix, where $v_i$ occupies the entry at row $r(v_i)$ and column $c(v_i)$ (see Figure 3.8). Is for this reason that we chose the term column for this function.

It is easy to see by definition that, when traversing any path $P$ of $S^-$, the vertex $w$ that follows $u$ has $c(w) \geq c(u)$, where equality happens only when $u \to w$ is a nose. This fact is fundamental in the geometrical argument of the next lemma that proves that "crossing" paths in the pictorial draw of $S^-$ share a common vertex. For the next lemma, let $C^-(v) = \min\{c(w) \mid r(w) = r(v)\}$ and $C^+(v) = \max\{c(w) \mid r(w) = r(v)\}$ for every $v \in V(S^-)$. That is, $C^-(v)$ and $C^+(v)$ are the minimum and maximum columns for the row that $v$ occupies

**Lemma 6.** Let $P = u_1, \ldots, u_j$ and $Q = w_1, \ldots, w_k$ be paths in $S^-$ such that

- $r(w_1) \leq r(u_1)$, and $r(w_k) \geq r(u_j)$,
- $c(u_1) = C^-(u_1)$, and $C(w_1) = C^-(w_1)$, and
- $c(u_j) = C^+(u_j)$, and $c(w_k) = C^+(w_k)$.

Then, some vertex of $S^-$ belongs to both paths.

*Proof.* Having defined $r$ and $c$ we can relate both, by thinking of the pair $(c(v) + \varepsilon r(v), r(v))$ (for a small enough $\varepsilon$) as the *coordinate* of the vertex $v$ in the plane, for every $v \in S^-$. Call $\mathrm{Gr}(P)$ to the graph of the continuous function that result from joining the coordinates of $u_i$ and $u_{i+1}$ $(1 \leq i < j)$ with a straight line. Note that $\mathrm{Gr}(P)$ is well defined as a function on $\mathbb{R} \to \mathbb{R}$ because $c(u_{i+1}) + \varepsilon r(u_{i+1}) > c(u_i) + \varepsilon r(u_i)$ for every $1 \leq i < j$ (see Figure 3.6). Analogously, $\mathrm{Gr}(Q)$ is the graph of the continuous function that result from joining the coordinates of $w_i$ and $w_{i+1}$ $(1 \leq i < k)$ with a straight line. So, $\mathrm{Gr}(P)$ goes from $(c(u_1) + \varepsilon r(u_1), r(u_1))$ to $(c(u_j) + \varepsilon r(u_j), r(u_j))$ and $\mathrm{Gr}(Q)$ goes from $(c(w_1) + \varepsilon r(w_1), r(w_1))$ to $(c(w_k) + \varepsilon r(w_k), r(w_k))$. Since $r(u_{i+1}) = r(u_i) \pm 1$ for every $i = 1, \ldots, j-1$, then $\mathrm{Gr}(P)$ and $\mathrm{Gr}(Q)$ must intersect according to the hypothesis of the lemma.

Let $x$ be the leftmost intersection point, and suppose $x$ is not the coordinate of some vertex in $S^-$. Then, there exist $u_a$ and $w_b$ such that $x$ belongs to the line segments of $\mathrm{Gr}(P)$ and $\mathrm{Gr}(Q)$ corresponding to $(u_a, u_{a+1})$ and $(w_b, w_{b+1})$, respectively. Recall once

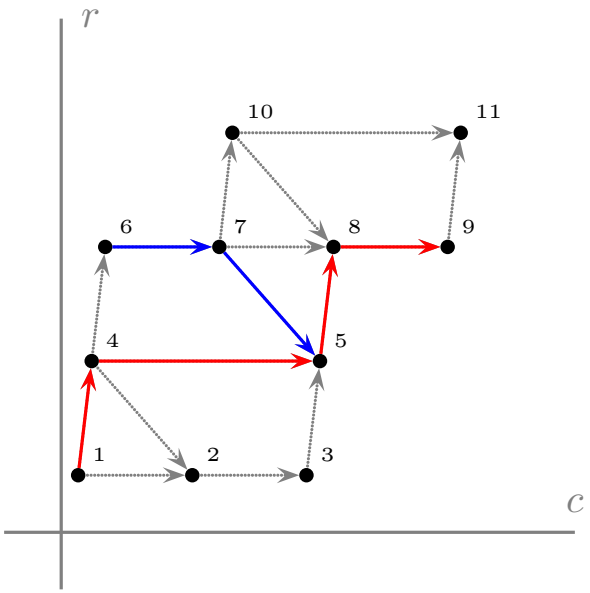



**Figure 3.6** – *The representation in the plane by the* coordinates *of Lemma 6 for the model in Figure 3.5. Paths $P$ and $Q$ intersect at $v_5$.*

again that $r(u_{a+1})$ is either $r(u_a)$ or $r(u_a) \pm 1$, while $r(w_{b+1}) \in \{r(w_b), r(w_b) \pm 1\}$. Then, the only possibility is that $r(u_{a+1}) = r(u_a) + 1$ and $r(w_b) = r(w_{b+1}) + 1$ (see Figure 3.7). Moreover, $u_a = v_i$, $w_{b+1} = v_{i+1}$, $w_b = v_j$, and $u_{a+1} = v_{j+1}$. Therefore, since $S^-$ contains no $\pi$-steps, we obtain that $u_a \to u_{a+1}$ is a nose, while $w_b \to w_{b+1}$ is a hollow. But this is impossible because $t_i s_{j+1}$ and $s_j t_{i+1}$ would be consecutive, while $t_i < t_{i+1}$ and $s_j < s_{j+1}$. □

**Lemma 7.** Every cycle $C$ in $S(\mathcal{I})$ contains exactly one $\pi$-edge.

*Proof.* By Lemma 5 we know that $C$ contains at least one $\pi$-edge. Suppose that it contains at least two and let $w_k \to u_1$ be the $\pi$-edge such that $r(u_1) > r(v_j)$ for every $\pi$-edge $v_i \to v_j$ in $C$. Then, $C$ contains two paths $P = u_1, \ldots, u_j$ and $W = w_1, \ldots, w_k$ that contain no $\pi$-edges and such that $u_j \to u_{j+1}$ and $w_0 \to w_1$ are $\pi$-edges (perhaps $u_{j+1} = w_1$ and $w_0 = u_j$). Moreover, since $r(u_j)$ equals either $r(u_{j+1})$ (when $u_j \to u_{j+1}$ is a $\pi$-hollow) or $r(u_{j+1}) - 1$ (when $u_j \to u_{j+1}$ is a $\pi$-step), then $r(u_j) \leq r(u_1)$. Then, since $r(w_k)$ equals either $r(u_1)$ or $r(u_1) - 1$, we conclude that $r(u_j) \leq r(w_k)$. That is, we are under hypothesis of Lemma 6, and so $C$ has a repeated vertex which contradicts the fact that $C$ is a cycle. □

Now we know that every cycle $C$ in $S(\mathcal{I})$ if formed by a path $P = w_1, \ldots, w_p$ in $S^-$ plus a $\pi$-edge $w_p \to w_1$. Such a path $P$ is said to be a *hollow p-ending path* or *step*

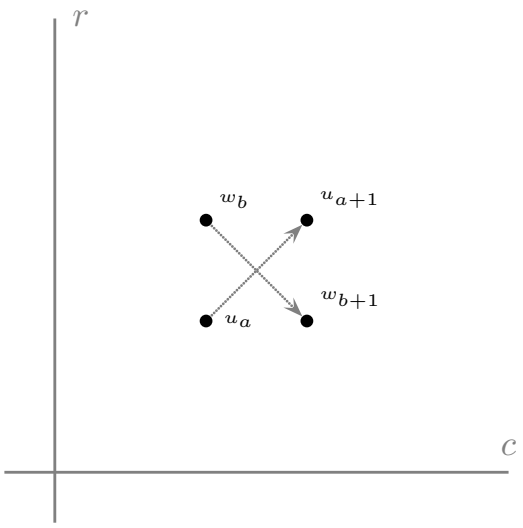



**Figure 3.7** – *Paths $P$ and $Q$ intersecting at some point $x$ that does not correspond to the coordinate of a vertex in $S^-$.*

*$p$-ending path* of $S^-$ according to whether $w_p \to w_1$ is a hollow or step, respectively. Let $L_0$ be the length constraints $(0, \nu, \eta, \sigma)$, and

- $U = \{i \mid v_i \to v_j \text{ is a nose of } P\}$,
- $D = \{i \mid v_i \to v_j \text{ is a non-}\pi \text{ hollow of } P\}$,
- $T = \{i \mid v_i \to v_j \text{ is a non-}\pi \text{ step of } P\}$.

Note that, since every nose increases $r$ by 1, every non-$\pi$-hollow decreases $r$ by 1, and every non-$\pi$-step preserves $r$, it follows that 1. $|U| = |D|$ if $v_p \to v_1$ is a hollow, and 2. $|U| + 1 = |D|$ if $v_p \to v_1$ is a step. Then, if $v_p \to v_1$ is a hollow, we obtain that the $L$-weight of $C$ is

$$\begin{aligned} L(C) =& L(P) - \ell + \eta(p) \\ =& \sum_{i \in U} (\ell + \nu(i)) + \sum_{i \in D} (-\ell + \eta(i)) + \sum_{i \in T} \sigma(i) - \ell + \eta(p) \\ =& \sum_{i \in U} \nu(i) + \sum_{i \in D} \eta(i) + \sum_{i \in T} \sigma(i) - \ell + \eta(p) \\ =& L_0(P) - \ell + \eta(p) \end{aligned} \tag{3.1}$$

Similarly, if $v_p \to v_1$ is a nose, then

$$L(C) = L(P) - \ell + \sigma(p) = L_0(P) - \ell + \sigma(p) \tag{3.2}$$

Then, by (3.1) and (3.2), the minimum possible value of $\ell$ such that $|C| \leq 0$ for every cycle $C$ is

$$\ell_* = \max \left\{ \begin{array}{l} \max\{L_0(P) + \eta(p) \mid P \text{ is a hollow } p\text{-ending path of } S^-\}, \\ \max\{L_0(P) + \sigma(p) \mid P \text{ is a step } p\text{-ending path of } S^-\} \end{array} \right\} \tag{3.3}$$

We remark that the length constraint $L_0$ above is only used for the sake of exposition. The fact that $L(P) = L_0(P)$ implied by (3.1) and (3.2) is actually showing that $L(P)$ depends not on the value of $\ell$. Indeed, each nose $v_i \to v_j$ has $L_0$-weight $\nu(i)$, each hollow $v_i \to v_j$ has $L_0$-weight $\eta(i)$, and every step $v_i \to v_{i+1}$ has $L_0$-weight $\sigma(i)$.

Equation (3.3) proves that there is always a minimum length $\ell_*$ for every $L$-constraint in which $\ell$ is an indeterminate value. The algorithm to find $\ell_*$ is quite simple; it suffices to find $L_0(P)+\eta(p)$ and $L_0(P)+\sigma(p)$ for every hollow and step $p$-ending path, respectively. Since $S^-$ is acyclic, we can find the weights of both $p$-ending paths in $O(n)$ time for every $w_p$. Then, taking into account that $O(n)$ such paths exist, the algorithm requires $O(n^2)$ time.

Finally, once $\ell_*$ is found, we can compute $\mathcal{U}_* = \mathcal{U}_{L_*}$ in $O(n^2)$ using Theorem 4, where $L_* = (\ell_*, \nu, \eta, \sigma)$. We claim that $\mathcal{U}_* = \{(s_i^*, t_i^*) \mid 1 \leq i \leq n\}$ is *minimal* in the sense defined by Pirlot in [13]. To see why, suppose $\mathcal{I} = \{(s_i, t_i) \mid 1 \leq i \leq n\}$ is an $L$-UIG model for some $\ell \geq \ell_*$, and consider some $i \in \{1, \ldots, n\}$. We saw in Section 3.1 that $s_i \geq d_L(v_1, v_i)$. Observe that if $P$ is a path from $v_1$ to $v_i$ in $S$ with $L(P) = d_L(v_1, v_i)$, then, by Lemma 5, the number of noses $u$ in $P$ must be greater than or equal to the number of hollows $d$ in $P$. Consequently,

$$s_i \geq d_L(v_1, v_i) = \ell(u - d) + L_0(P);$$

as this equation holds also for $L_*$, we conclude that $s_i^* \leq s_i$. That is,

(i) $\ell_* \leq \ell$, and

(ii) $s_i^* \leq s_i$

for every $L$-UIG model $\mathcal{I}$ equivalent $\mathcal{U}_*$. Conditions (i) and (ii) are exactly the same as in Pirlot's definition of minimal models. Thus, it makes sense to say that $\mathcal{I}$ is a *minimal $L$-UIG model* when it satisfies (i) and (ii) for every equivalent $L$-UIG model. The main theorem of this thesis then follows.

**Theorem 8.** Every PIG model is equivalent to a minimal $L$-UIG model for every length constraints $L$ in which $\ell$ is an indeterminate. Moreover, such a model is equal to $\mathcal{U}_{(\ell_*, \nu, \eta, \sigma)}$, where $\ell_*$ is as in (3.3), and it can be obtained in $O(n^2)$ time.

## 3.3 Short but not minimal models

In [11], Mitas represents the synthetic graph $S^-(\mathcal{I})$ in a matricial form as in Figure 3.8. In this picture each vertex represents a vertex $v_i$ of $S$ that is drawn occupying the coordinate $(r(v_i), c(v_i))$ of the plane (as in Lemma 6), while each edge corresponds to a nose (vertical solid arrows) or a non-$\pi$ hollow (diagonal dashed arrows). (We can think that the missing steps are implicit.)

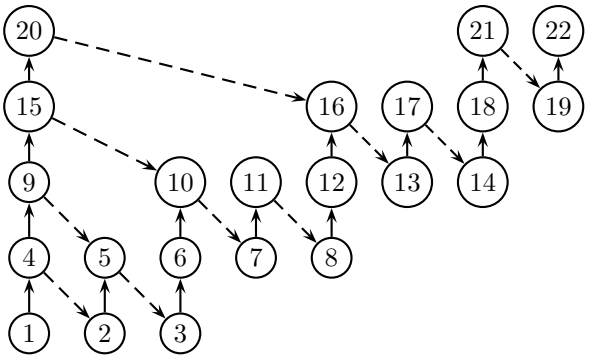



**Figure 3.8** – *Mitas matricial representation of the synthesis of a reduced PIG model (semiorder).*

Mitas' goal is to find a minimal UIG model $\mathcal{U}$ equivalent to $\mathcal{I}$ with no separation constraints for the extremes of $\mathcal{U}$, except those that are required for $\mathcal{U}$ to be equivalent to $\mathcal{I}$. In other words, Mitas considers only the case in which $\sigma = \eta = \nu = 1$[1]. Moreover, Mitas requires the input model $\mathcal{I}$ to be *reduced*. This means that either $s_i$ and $s_{i+1}$ or $t_i$ and $t_{i+1}$ are not consecutive in $\mathcal{I}$ for every $1 \leq i \leq n$. In terms of the synthetic graph, this means that there is either a nose or hollow going from $v_i$ and a nose or hollow ending at $v_i$. This is the reason why Mitas does not draw the step edges, as they are redundant for this problem.

Mitas asserts that the minimum length $\ell$ needed to represent $\mathcal{I}$ with length constraints $L = (\ell, 1, 1, 1)$ is $\ell = \max\{d_L(v_1, v_j) - d_L(v_1, v_i) \mid r(v_i) = r(v_j)\} + 1$, where $d_L$ is calculated on $S^-$. For the example in Figure 3.8 this length is $\ell = d_L(v_1, v_{19}) - d_L(v_1, v_{15}) = d(v_1, v_{22}) - d(v_1, v_{20}) = 15$. But we can see in Figure 3.9 that $\mathcal{I}$ can be represented with a shorter model $\mathcal{U}$ of length $d = 13$. In fact, the path from $v_9$ to $v_{14}$ of length 12 (regardless the value of $\ell$, recall (3.1)) has maximum $L$-weight among the hollow ending paths, while the path from $v_{15}$ to $v_{14}$ of length 9 has maximum $L$-weight among the step ending paths. Since $\eta = 1$, (3.3) guaranties that $\mathcal{U}$ is a minimal model.

[1] Actually, Mitas works with closed intervals and different intervals of $\mathcal{I}$ could share the same extreme. This is not important, though, as it is only a matter of output. The input for both problems are equivalent and the outputs are in a one-to-one correspondence. Thus, the counterexample of this section disproves this case as well.

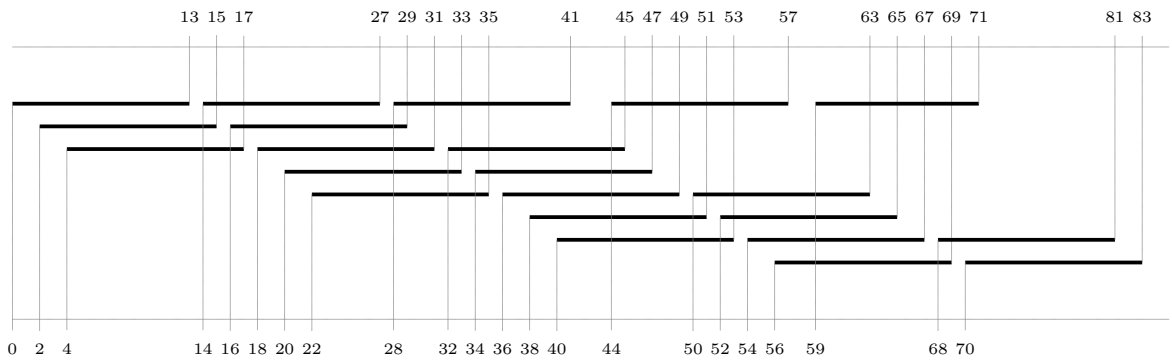



**Figure 3.9** – *Reduced interval model of length 13.*

The problem with the previous approach is that a path of length $\max\{d(v_1, v_j) - d(v_1, v_i)\}$ from $v_i$ to $v_j$ could not exist. Mitas claims this to be true in Theorem 5 of [11] and, under the assumption that this is true, the proposed value is correct. In [12] there is a review of Mitas' algorithm that neither addresses this problem. We remark that, despite this problem, Mitas algorithm outputs a UIG model equivalent to $\mathcal{I}$ with length $\ell < n$.

## 3.4 Powers of paths

Say that a UIG model $\mathcal{U}$ is *odd* when its length $\ell$ is odd and every beginning point is even. It is not hard to see that every PIG model $\mathcal{I}$ is equivalent to an odd UIG model. Indeed, by Theorem 1, $\mathcal{I}$ is equivalent to a UIG model $\mathcal{U}' = \{(s_i, s_i + \ell) \mid 1 \leq i \leq n\}$ for some length $\ell$. Clearly, the model $\mathcal{U} = \{(2s_i, 2(s_i + \ell) + 1 \mid 1 \leq i \leq n\}$ is odd and equivalent to $\mathcal{U}$. In this section we apply the synthetic graph of $\mathcal{I}$ so as to find the odd UIG model equivalent to $\mathcal{I}$ with minimum interval length.

The reason for studying this particular problem has to do with the induced subgraphs of powers of paths. Recall that, in an analogy to Pirlot's definition for minimal UIG models, we refer to $P_q^k$ as the *minimal extension* of $\mathcal{I}$ when $G(\mathcal{I})$ is an induced subgraph of $P_q^k$ and:

- $k \leq j$, and
- $q \leq r$

for every $P_r^j$ such that $G(\mathcal{I})$ is an induced subgraph of $P_r^j$.

Clearly, $P_q^k$ is a UIG graph for any value of $q$ and $k$, as it is the graph represented by the odd UIG model $\mathcal{P}_q^k = \{(2i, 2(i+k)+1) \mid 1 \leq i \leq q\}$. In particular, note that the interval length of $\mathcal{P}_q^k$ is minimum among all the odd UIG models that represent $P_q^k$. By definition, every subset $\mathcal{U}$ of $\mathcal{P}_q^k$ is an odd UIG model. Conversely, every odd UIG model is a subset of $\mathcal{P}_q^k$, for some appropriate values of $q$ and $k$, because every even $p \in \mathbb{N}$ is a beginning point of some interval in $\mathcal{P}_q^k$. Thus, every PIG graph is an induced subgraph of a power of a path; this fact was first noted in [10]. Furthermore, the problem of finding the minimal extension of $\mathcal{I}$ corresponds to the problem of finding the minimum odd UIG model $\mathcal{U}$ equivalent to $\mathcal{I}$.

In [4], Costa et al. developed an $O(n^2)$ time algorithm to obtain the minimal extension of a PIG model $\mathcal{I}$. Their algorithm explicitly stores all the vertices of $P_q^k$, and it is not easy to follow. We can solve the problem in $O(n^2)$ time by simply observing that $\mathcal{U} = \mathcal{U}_L(\mathcal{I})$ is an odd model equivalent to $\mathcal{I}$ (see Theorem 9 below), for $L = (\ell, \nu = 1, \eta = 1, \sigma = 2)$, where $\ell$ is taken as in (3.3). Then, since $s_{i+1} - s_i \geq \sigma = 2$ and $s_n \geq d_L(v_1, v_n)$ in every odd model equivalent to $\mathcal{I}$, the model $\mathcal{U}$ so obtained represents the minimal extension $P_q^k$ of $\mathcal{I}$.

**Theorem 9.** If $\mathcal{I}$ is a PIG model, then $\mathcal{U}_L(\mathcal{I})$ is an odd UIG model for $L = (\ell_*, \nu = 1, \eta = 1, \sigma = 2)$, where $\ell_*$ is taken as in (3.3).

*Proof.* By definition, we have to prove that $\ell_*$ is odd and every beginning point of $\mathcal{U} = \mathcal{U}_L(\mathcal{I})$ is even.

To prove that $\ell_*$ is odd, consider any cycle $C = w_1, \ldots, w_q, w_1$ of maximum $L$-weight, and let $u$, $d$, and $e$ be its number of noses, hollows, and steps, respectively. Recall that $u = d - 1$, as it was shown while deriving (3.1) and 3.2. Hence, $0 = L(C) = -\ell_* + 2(u + e) + 1$ and $\ell_*$ is odd as desired.

To see that every beginning point of $\mathcal{U}$ is even, just recall that every nose has weight $\ell_* + 1$, every hollow has weight $1 - \ell_*$ and every step has weight 2. Then, since $\ell_*$ is odd, it follows that all the edges of $S$ have even $L$-weight, thus every path has an even $L$-weight. Then, since every beginning point is a the $L$-distance of two vertices in $S$, the result follows. □

One nice consequence of Theorem 9 is that it certifies that a minimal extension $P_q^k$ of $\mathcal{I}$ always exists. In [4], the authors constantly state that they find the minimum $k$ and **then**, for this fixed $k$ , they find the minimum $q$. In fact, by Theorem 8, it matters not in which order they are found, and it is for this reason that we refer to the output in [4] as a minimal extension.

# 4 Conclusions and open problems

In this work we solved the problem of finding a UIG model with integer extremes and minimum interval length that is equivalent to an input PIG model $\mathcal{I}$. Moreover, the model so obtained satisfies a set of separation constraints that are also given as input. To solve this problem, we developed an algorithm that decides if there exists a model satisfying a quadruple $L$ of length constraints. This algorithm works by querying if the synthetic graph of $\mathcal{I}$ has a cycle of positive $L$-weight. Then, we devise a second algorithm that finds the minimum feasible length $\ell$, again exploiting the synthetic graph structure. We think that this solution can be extended so as to solve the more general problem of finding a unit circular-arc (UCA) model with integer extremes and minimum length when a proper circular-arc (PCA) model is given as input. Circular-arc models and graphs are the generalization of interval models and graphs, where the real line is replaced by a circle. That is, two vertices of a circular-arc graph are adjacent if and only if their corresponding arcs on the circle intersect. As it happens with interval graphs, PCA graphs are obtained when no arc is properly contained in other arcs, while UCA graphs are obtained when all the arcs have the same length. That is, CA is to IG what PIG is to PCA and UIG is to UCA. As we said, PIG is a subclass of PCA; indeed, any PIG model can be embedded in a sufficiently large circle. Because of this, it makes sense to define synthetic graphs for PCA models so as to impose the separation constraints that its arc must satisfy in order to be a UCA model. The major inconvenient about this approach is that some PCA models admit no equivalent UCA models. Thus, for this approach to work, the recognition problem for UCA graphs must be solved in this framework.

**Problem 1.** Give an algorithm to find a UCA model of minimum length equivalent to an input PCA model, if such a model exists.

Our algorithm that finds the minimum $\ell_*$ such that $\mathcal{I}$ is equivalent to a UIG model of length $\ell_*$ requires $O(n^2)$ time. The question that naturally arises is if this problem can be solved in linear time. To find $\ell_*$, our algorithm finds the maximum weight path of the synthetic graph that goes from the leftmost vertex in row $r$ to the rightmost vertex in row $r$, for every row $r$. In fact, all we need is the maximum of such paths. Maybe it is possible to find the corresponding row more efficiently. A related problem is how to build the model of length $\ell_*$ in $o(n^2)$ time once $\ell_*$ has been found.

**Problem 2.** Design an $o(n^2)$ time algorithm to find the minimum value $\ell_*$ such that an input PIG model is equivalent to a UIG model with integer extremes and interval length $\ell_*$.

**Problem 3.** Give an $o(n^2)$ time algorithm that, given a length $\ell$, outputs a PIG model with integer extremes and interval length $\ell$, if such a model exists.

Another thing that came to our minds while writing this thesis is how the constraints rules can vary to generate other problems similar to the one in hand. In this document we require each beginning point to be separated from the next one *at least* by a given distance. This is expressed in terms of some natural functions $\sigma$, $\nu$ and $\eta$. It makes sense to consider, also, the problem where some separation constraints are given by equalities as well. Then, we could express that some beginning points must be separated from the next beginning point by *exactly* a given distance, while others must be separated by *at least* or *at most* some given distance. Another example is to consider general rules for separation: each pair of extremes must be separated by exactly (at most) some given distance. In this case, a solution needs not exist.

**Problem 4.** Study different sets of rules for the separation constraints.

Finally, in this thesis we work with interval models in which no pair of extremes share the same point. Under this constraint, it is not important if intervals are open or closed. In other works (e.g Mitas [11]) the authors allow representations with overlapping extremes. In this case, you must choose between open and closed intervals representations.

**Remark 1.** All the algorithms work when the extremes of the output model are not restricted to be different. All we need to do is to adapt the values of constraints imposed by $\sigma$, $\nu$, and $\eta$, depending on whether the intervals are all open or all closed.

# 5 Bibliography


[1] Kenneth P. Bogart and Douglas B. West. A short proof that "proper = unit". *Discrete Math.*, 201(1-3):21–23, 1999. ISSN 0012-365X.

[2] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. MIT Press, Cambridge, MA, third edition, 2009. ISBN 978-0-262-03384-8.

[3] Derek G. Corneil, Hiryoung Kim, Sridhar Natarajan, Stephan Olariu, and Alan P. Sprague. Simple linear time recognition of unit interval graphs. *Inform. Process. Lett.*, 55(2):99–104, 1995. ISSN 0020-0190. doi: 10.1016/0020-0190(95)00046-F. URL http://dx.doi.org/10.1016/0020-0190(95)00046-F.

[4] Vítor Costa, Simone Dantas, David Sankoff, and Ximing Xu. Gene clusters as intersections of powers of paths. *J. Braz. Comput. Soc.*, 18(2):129–136, 2012. ISSN 0104-6500. doi: 10.1007/s13173-012-0064-8. URL http://dx.doi.org/10.1007/s13173-012-0064-8.

[5] Xiaotie Deng, Pavol Hell, and Jing Huang. Linear-time representation algorithms for proper circular-arc graphs and proper interval graphs. *SIAM J. Comput.*, 25 (2):390–403, 1996. ISSN 0097-5397. doi: 10.1137/S0097539792269095. URL http://dx.doi.org/10.1137/S0097539792269095.

[6] Frédéric Gardi. The Roberts characterization of proper and unit interval graphs. *Discrete Math.*, 307(22):2906–2908, 2007. ISSN 0012-365X.

[7] Martin Charles Golumbic. *Algorithmic graph theory and perfect graphs*, volume 57 of *Annals of Discrete Mathematics*. Elsevier Science B.V., Amsterdam, second edition, 2004. ISBN 0-444-51530-5. With a foreword by Claude Berge.

[8] Min Chih Lin, Francisco J. Soulignac, and Jayme L. Szwarcfiter. A simple linear time algorithm for the isomorphism problem on proper circular-arc graphs. In *Algorithm theory—SWAT 2008*, volume 5124 of *Lecture Notes in Comput. Sci.*, pages 355–366. Springer, Berlin, 2008. doi: 10.1007/978-3-540-69903-3_32. URL http://dx.doi.org/10.1007/978-3-540-69903-3_32.

[9] Min Chih Lin, Francisco J. Soulignac, and Jayme L. Szwarcfiter. Short models for unit interval graphs. In *LAGOS'09—V Latin-American Algorithms, Graphs and Optimization Symposium*, volume 35 of *Electron. Notes Discrete Math.*, pages 247–255. Elsevier Sci. B. V., Amsterdam, 2009.

[10] Min Chih Lin, Dieter Rautenbach, Francisco Juan Soulignac, and Jayme Luiz Szwarcfiter. Powers of cycles, powers of paths, and distance graphs. *Discrete Applied Mathematics*, 159(7):621 – 627, 2011. ISSN 0166-218X. doi: DOI:10.1016/j.dam.2010.03.012. URL http://www.sciencedirect.com/science/article/B6TYW-4YX0BGM-2/2/2dbfafd2caf6e3a6bd4da025fef9cd69. Graphs, Algorithms, and Their Applications – in Honor of Martin Charles Golumbic on the Occasion of His 60th Birthday.

[11] Jutta Mitas. Minimal representation of semiorders with intervals of same length. In Vincent Bouchitté and Michel Morvan, editors, *Orders, algorithms, and applications (Lyon, 1994)*, volume 831 of *Lecture Notes in Comput. Sci.*, pages 162–175. Springer, Berlin, 1994.

[12] M. Pirlot and Ph. Vincke. *Semiorders*, volume 36 of *Theory and Decision Library. Series B: Mathematical and Statistical Methods*. Kluwer Academic Publishers Group, Dordrecht, 1997. ISBN 0-7923-4617-3. Properties, representations, applications.

[13] Marc Pirlot. Minimal representation of a semiorder. *Theory and Decision*, 28(2):109–141, 1990. ISSN 0040-5833. doi: 10.1007/BF00160932. URL http://dx.doi.org/10.1007/BF00160932.

[14] Fred S. Roberts. Indifference graphs. In *Proof Techniques in Graph Theory (Proc. Second Ann Arbor Graph Theory Conf., Ann Arbor, Mich., 1968)*, pages 139–146. Academic Press, New York, 1969.

[15] Dana Scott and Patrick Suppes. Foundational aspects of theories of measurement. *J. Symb. Logic*, 23:113–128, 1958. ISSN 0022-4812.