Models and Algorithms for Chronology

Gilles Geeraerts[1], Eythan Levy[2], and Frédéric Pluquet[3]

1 **Université libre de Bruxelles, Computer Science Department, Brussels, Belgium**
gigeerae@ulb.ac.be
2 **Tel Aviv University, Archaeology Department, Tel Aviv, Israel**
eythan.levy@gmail.com
3 **École Supérieure d'Informatique (HE2B-ESI), Brussels, Belgium**
fpluquet@he2b.be

Abstract

The last decades have seen the rise of many fundamental chronological debates in Old World archaeology, with far-reaching historical implications. Yet, outside of radiocarbon dating – where Bayesian formal tools and models are applied – these chronological debates are still relying on non-formal models, and dates are mostly derived by hand, without the use of mathematical or computational tools, albeit the large number of complex constraints to be taken into account. This article presents formal models and algorithms for encoding archaeologically-relevant chronological constraints, computing optimal chronologies in an automated way, and automatically checking for chronological properties of a given model.

1998 ACM Subject Classification J.2 [Physical Sciences and Engineering] Archaeology

Keywords and phrases Chronology, Algorithms, Archaeology, Formal methods

Digital Object Identifier 10.4230/LIPIcs.TIME.2017.13

1 Introduction

1.1 Motivation

Reconstructing ancient chronologies based on archaeological data is a complex task, which has haunted researchers since the start of the discipline. Current chronologies of given regions result from an integration of diverse data such as historical records (providing regnal dynasties for example), pottery sequences (providing relative chronologies for a given region), stratigraphic sequences (providing synchronisms through stratified pottery and artefacts, both local and imported), ancient inscriptions (providing synchronisms between different kings for example) and laboratory methods (providing date-ranges for a given event, through radiocarbon dating for example). In most cases, these data constitute a complex web of intricate information, connecting archaeological data of neighbouring sites and regions in a subtle way. Hence, any change in chronological hypotheses at one end of the network (say, changing the historical dates of a given king for example) can have far-reaching chronological repercussions throughout the network (say for example that inscriptions of this king have been found in a specific archaeological layer, bearing specific pottery types). How can we model such constraints formally, and how can we use our formal model to automatically derive a global chronology for a region of interest, through the integration of several scattered local chronological constraints? We have observed that, outside of carbon dating, where Bayesian statistical tools are used to formally derive date estimates [15], traditional chronological research still uses no formal model or algorithm to integrate sets of local chronological

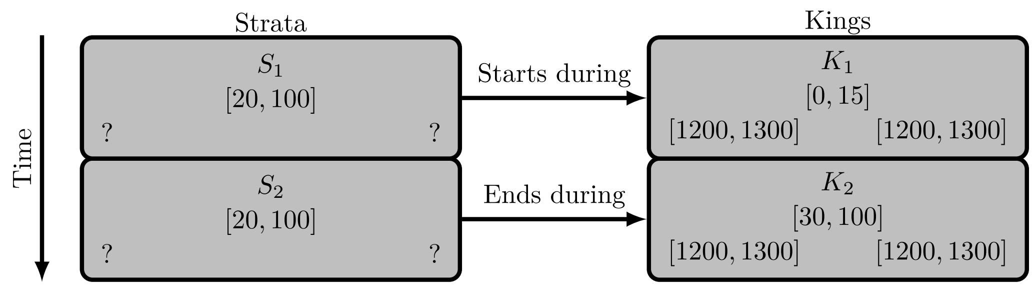

Figure 1 Our running example: Chronoland. Each period is represented by a rectangle containing the period's name. The bounds on each period's duration, start date and end date are displayed in the centre, bottom left corner and bottom right corner respectively.

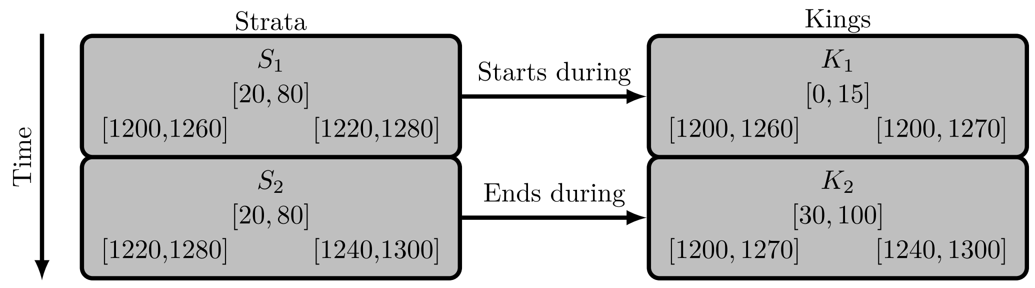

Figure 2 Optimal date and duration estimates for the Chronoland example.

constraints into a global chronology. In most cases (outside of radiocarbon dating), the date estimates proposed by researchers are derived by hand, thus certainly obtaining non-optimal results, since manual treatment of any set of chronological constraints of non-trivial size is almost impossible. The purpose of this paper is to present a model that allows one to formally model archaeologically-relevant chronological constraints, to compute optimal chronologies through specific algorithms, and to automatically check chronological properties of a given model. Applications of such models are numerous, since the past decades have seen the rise of many important – and still unsettled – chronological debates in Old World archaeology, with far-reaching historical implications. Important recent examples of such debates include the Thera Eruption debate, where two opposing chronologies differ by *more than a century*, with important implications regarding the chronology of the Aegean Bronze Age [8]. A second well-known example is the Iron Age Low Chronology debate for the southern Levant, where two chronologies differ as to whether the first full-fledged territorial states in the region are to be dated to the 10^{th} or the 9^{th} century BCE [13].

1.2 Running example

As a didactic example of the kind of data and constraints that typically appear in chronological debates, we will start with a small test case, in order to show that even small examples, with far less constraints than any real archaeological case study, are impractical to treat manually and require a formal and algorithmic approach. The test case is called Chronoland, and is presented hereinafter.

The story of Chronoland. In the kingdom of Chronoland, Kings K_1 and K_2 reigned in succession. We do not know their precise reign dates, but both reigns are known to have occurred between 1200 and 1300 CE. Furthermore, K_1's reign did not exceed 15 years, since we know he died at a rather young age, and King K_2 reigned for at least 30 complete years, since we have a monument dated to his 31st year[1]. We can also safely assume that K_2's reign did not exceed 100 years. Now Chronocity, the capital city of Chronoland, was excavated and only two strata were found, namely strata S_1 and S_2. We know that stratum S_1 was built by King K_1 since he thus claims in a stela found there in-situ, and we also know that Stratum S_2 ended during the reign of King K_2, since ancient annals tell us that the city was destroyed during his reign and never reoccupied. Finally, we assume that each one of our strata has a duration of at least 20 years and at most 100 years. The data of the example are given in schematic form in Figure 1, and several natural questions can be asked about it, such as:

1. What are the most precise estimates (ranges) we can gather for the start date, end date and duration of each king and stratum of Chronoland?
2. Has King K_1 built stratum S_2? (Recall that the data only asserts he has built S_1.)

Observe that the second question could admit three different answers because of the uncertainties on the dates: (i) 'yes', K_1 has surely built S_2, in all possible scenarios that fit with the constraints on the dates and durations; or (ii) 'no', K_1 has surely not built S_2 (again in all possible scenarios); or (iii) 'maybe', i.e. some dates that respect the constraints, imply that K_1 has built S_2, and some don't.

Chronology computation. We start by discussing the first question (optimal ranges). These optimal estimates are shown in Figure 2. One can easily see that they are not straightforward and require a close look at the model. Here are the steps that one could follow to obtain these results:

1. **Initialisation.** The date ranges of K_1 and K_2 can be improved using the duration estimates. The 30 years minimum duration of K_2 imply that K_2 starts in [1200,1270] and that it ends in [1230,1300].
2. **K_1–K_2 sequence.** The end of K_1 equals the start of K_2, hence it is in [1200,1270]. The start of K_1 is thus also in [1200,1270], since a start date cannot exceed its corresponding end date.
3. **S_1–K_1 synchronism.** The 'S_1 starts during K_1' synchronism implies that the start of S_1 is in [1200,1270]. The [20,100] duration range now implies that the end of S_1 is in [1220,1370].
4. **S_1–S_2 sequence.** The start of S_2 equals the end of S_1, hence it is in [1220,1370]. The [20,100] duration range now implies that the end of S_2 is in [1240,1470].
5. **S_2–K_2 synchronism.** The 'S_2 ends during K_2' synchronism implies that the end of S_2 is in [1240,1300]. The [20,100] duration range now implies that the start of S_2 is in [1220,1280]. The duration range can now be reduced to [20,80] since the earliest start of S_2 is 1220 and its latest end is 1300.
6. **S_1–S_2 sequence.** The end of S_1 equals the start of S_2, hence it is in [1220,1280]. The [20,100] duration range now implies that the start of S_1 is in [1200,1260]. The duration range can now be reduced to [20,80] since the earliest start of S1 is 1200 and its latest end is 1280.

[1] We use a cautious bound of 30 years instead of 31 years because we do not know if the king completed his 31st year of reign (i.e. he might have died in the course of his 31st year).

7. **S_1–K_1 synchronism.** We need to come back to the 'S_1 starts during K_1' synchronism. Our new upper bound of 1260 on S_1's start needs to be propagated back to K_1, implying a new improved range of [1200,1260] on K_1's start, instead of the former [1200,1270].
8. **S_2–K_2 synchronism**. Finally, we need to consider anew the 'S_2 ends during K_2' synchronism. Our new lower bound of 1240 on S_2's end needs to be propagated back to K_2, implying a new improved range of [1240,1300] on K_2's end, instead of the former [1230,1300].

Note the difficulty of manually solving this simple problem, especially the final 'retroaction' step, where K_1's start date and K_2's end date get further refined by 10 years thanks to their synchronisms with S_1 and S_2 respectively. This final step is somewhat unexpected since S_1 and S_2 had no a priori absolute chronology of their own (they had only duration estimates), and thus derived their absolute dates from K_1 and K_2. Now, let us come back to the question whether king K_1 has built stratum S_2. It turns out that this is impossible, because K_1's reign lasts at most 15 years, but at least 20 years separate the respective start dates of S_1 and S_2, since S_1 lasts at least 20 years. However, again, deriving this information from Figure 1 or Figure 2 is not straightforward. Also note that this example is of small dimensions compared to real archaeological data for which manual treatment of the information is practically impossible.

1.3 Contribution

As can be seen from the Chronoland example, rigorous and automatic techniques for reasoning about chronological problems are needed. To the best of our knowledge, no such techniques are available today, and the kind of reasoning we exhibit in the Chronoland example are performed 'by hand' by archaeologist, using restricted data sets. In this paper, **our first contribution is a formal constraint language (Section 2) that can express most relationships between periods which are needed for practical cases of archaeology**. More precisely, our language can express: (i) lower and upper bounds on the start dates, end dates and durations of periods; (ii) sequences (such as period S_2 follows directly S_1); and (iii) different kinds of synchronisms between periods (such as 'S_1 starts during K_1', etc). Then, **our second contribution consists of several algorithmic techniques (Section 3) to manipulate such constraints and extract information that is meaningful to archaeologists**, namely: (i) what are the *tightest bounds* on the start dates, end dates and durations of all periods that one can infer from a given set of constraints? (ii) is the set of constraints *satisfiable*. That is, is it possible to assign dates to the starts and ends of all periods that satisfy all the constraints? (iii) do the constraints *imply* that two given periods have a non-empty intersection? That is, is the intersection guaranteed to exist, for all choices of start and end dates of the periods that satisfy the constraints? We call this problem the *sure-contemporaneity* problem; and, finally (iv) do the constraints *allow* a non-empty intersection between two periods? That is, does there *exist* a choice of dates for the starts and ends of all periods that satisfy the constraints and where the two given periods intersect? We call this problem the *possible-contemporaneity* problem. Observe that both the sure- and possible-contemporaneity problems can be invoked to answer our question: 'Has K_1 built S_2?'

Following [17], we translate our constraints in directed weighted graphs, and reduce the computation of the tightest bounds to an all-pair shortest paths computation in this graph (and, as particular case, we obtain an algorithm for the satisfiability problem which amounts to detecting negative cycles in the graph). Such computation can be carried out *in polynomial time* (in the number of periods) using classical algorithms [12], which is a clear strong point of approach, enabling scaling to examples of big dimensions. Since all-pairs shortest paths algorithms also detect negative cycle, we can also test satisfiability in polynomial time[2]. Finally, we show that, after computation of the shortest paths, the sure- and possible-contemporaneity problems can be solved *in constant time.*

1.4 Related works

The best known use of computer-assisted techniques in Archaeology is in a setting which is different from ours, namely the calibration of radiocarbon dates [15] thanks to the OxCal software. The aim of Oxcal is to refine estimates of dates computed from radiocarbon samples, by taking into account extra information such as the order of strata. Due to the stochastic nature of radiocarbon decay, these methods are probabilistic (they rely on Bayes's theorem) while our methods are not.

Concerning relative chronology, other formal approaches exist, based on the Harris matrix [7] or the generalised schemes of Sharon [16] and Holst [9]. These works differ from ours in several way. First, they aim at at obtaining a feasible sorting of the archaeological features, but do not estimate their absolute time-frame, while our method provides both *absolute* and *relative* information on the periods. Second, the underlying computational problem they address is NP-complete (hence, they rely on heuristics), while we have identified problems that can be solved in polynomial time. In fact, an early paper by Kromholz [11] already combined some relative and absolute dating elements, though in a limited way, through the application of business-oriented tools (such as Pert charts) to chronological problems. We are not aware however of any later paper that took on this novel approach.

On the other hand, the constraints we rely on are actually a special case of the *zone* data structure developed in the framework of *timed automata* [1, 6, 4] that form the cornerstone of efficient tools for the analysis of timed automata such as UPPAAL [2] and TiAMo [5]. In our case, however, variables take natural values, instead of real values in the case of timed automata.q Our constraints are also a particular case of general classes of constraints developed in the artificial intelligence community for temporal reasoning, see for instance [14]. Those constraints are more expressive than ours and basic problems about them are already NP-hard [14] while we propose polynomial-time algorithms.

2 Modelling chronology problems

In this section, we introduce our model for the chronology problems we have sketched in the introduction. Throughout the paper, we denote the set of natural and integers numbers by $\mathbb{N}$ and $\mathbb{Z}$ respectively. We consider closed interval on $\mathbb{N}$ (i.e., convex subsets of $\mathbb{N}$), with natural endpoints. We use the usual notation $[a, b]$ to denote intervals, and for $I = [a, b]$, we let $\ell(I) = a$ and $u(I) = b$.

2.1 Periods and chronologies

Periods. We fix a finite set $\mathcal{P}$ of *periods.* We use the general term 'period' to speak about continuous periods of time characterised by a start date, an end date and a duration (defined

[2] Similar techniques are used in the setting of timed automata, see related works.

as the difference between the end and start dates)[3]. Examples of periods include historical eras ('The Middle Ages'), archaeological strata, king's reigns and ceramic periods, among others.

Chronologies. A *chronology* $\mathcal{C}$ on a set of periods $\mathcal{P}$ is function that associates, to each period $p \in \mathcal{P}$, an interval $\mathcal{C}(p) = [a_p, b_p] \subset \mathbb{N}$ with natural endpoints a_p and b_p. For all periods p, the interval $\mathcal{C}(p)$ represents the whole duration of the period, in the sense that its endpoints represent the start and the end *dates* of the period. Throughout the paper, we assume that all dates are given simply as years, but a finer granularity can be used if need be (for example, dates can represent days). Observe that, for historical events, start and end dates could be negative, but we assume that there is a lower bound on all dates[4] (i.e., an 'origin of time'), which we identify with 0 (hence we have $a_e, b_e \geq 0$ for all $e \in \mathcal{P}$). A chronology $\mathcal{C}$ for our running example could be s.t. $\mathcal{C}(K_1) = [1210, 1222]$, which is compatible with the constraints in Figure 1.

2.2 Constraints

Common chronological constraints. Our goal is to formalise a large set of chronological constraints relevant to archaeology and history. In these fields, the most common constraints can be grouped in three families:

Bounds. The first family consists of lower and upper bounds on the start date, end date and duration of a period. For example, in Figure 1, the $[20, 100]$ years constraint on S_1's duration, or the $[1200, 1300]$ constraint on K_1's start date belong to this family.

Sequences. The second family expresses that a certain period starts where the preceding period ends, as in the K_1, K_2 sequence, and the S_1, S_2 sequence.

Synchronisms. Finally, the third family expresses diverse sorts of *synchronisms*, such as contemporaneity (two periods having a non-empty intersection, as in two contemporary kings), 'starts during' (as in 'S_1 starts during K_1) and 'ends during' (as in 'S_2 ends during K_2').

Let us now discuss a formal constraint language that allows us to describe all these constraints.

A formal model of constraints. To all finite sets of periods $\mathcal{P} = \{p_1, \ldots, p_n\}$, we associate a set of *variables* $\mathcal{V}(\mathcal{P}) = \{z_0, \mathsf{beg}(p_1), \mathsf{end}(p_1), \ldots \mathsf{beg}(p_n), \mathsf{end}(p_n)\}$, interpreted over the integers. For each period p_i, variables $\mathsf{beg}(p_i)$ and $\mathsf{end}(p_i)$ represent respectively the beginning and end of p_i. Variable z_0 is a special variable that is assumed to be always equal to 0 and its purpose will become clear later. Then, an atomic constraint on $\mathcal{P}$ is an expression of one of the following forms: either $x - y \sim c$ or $x \sim c$, where x, y are variables from $\mathcal{V}(\mathcal{P})$, $c \in \mathbb{Z} \cup \{+\infty\}$, and $\sim$ is either $\leq$ or $\geq$ or $=$. Finally, a *constraint* is a finite *conjunction* of atomic constraints[5]. For an atomic constraint ψ and a constraint φ, we write $\psi \in \varphi$ iff ψ is a conjunct of φ.

[3] Punctual events could be seen as special cases of periods have equal start and end dates, and null duration.
[4] In practice, this is not restrictive since events can always be associated to an epoch, even if this is very broad.
[5] Observe that neither disjunction nor negation are allowed.

Semantics. Let us now define the semantics of constraints in terms of chronologies. Intuitively, a constraint is meant to define a set of *possible chronologies*, which are all those that are compatible with the constraint. Formally, a chronology $\mathcal{C}$ (on set of periods $\mathcal{P}$) *satisfies* an atomic constraint $x - y \sim c$ (respectively, $x \sim c$) iff $\nu(x) - \nu(y) \sim c$ (respectively, $\nu(x) \sim c$), where $\nu : \mathcal{V}(\mathcal{P}) \to \mathbb{N}$ is the function associating to each variable x a *valuation* according to $\mathcal{C}$, i.e., for all $x \in \mathcal{V}(\mathcal{P})$: (i) $\nu(x) = 0$ if $x = z_0$; (ii) $\nu(x) = \ell(\mathcal{C}(e))$ if $x = \mathsf{beg}(e)$; and (iii) $\nu(x) = u(\mathcal{C}(e))$ if $x = \mathsf{end}(e)$. When a chronology $\mathcal{C}$ satisfies an atomic constraint ψ, we write $\mathcal{C} \models \psi$. We extend this notion of satisfaction to constraints: $\mathcal{C}$ satisfies a constraint $\varphi = \psi_1 \wedge \psi_2 \wedge \cdots \wedge \psi_n$ (noted $\mathcal{C} \models \varphi$) iff $\mathcal{C}$ satisfies all conjuncts of φ, i.e., $\mathcal{C} \models \psi_i$ for all $1 \leq i \leq n$. We denote by $[\![\varphi]\!]$ the set of all chronologies $\mathcal{C}$ s.t. $\mathcal{C} \models \varphi$. Remark that this set could be empty, for instance if we have specified constraints that are not satisfiable. Note also that two different constraints can encode the same chronologies, e.g. when there are redundant atomic constraints. For example, $\varphi = x \geq 0 \wedge x \leq 1 \wedge y \geq 0 \wedge y \leq 1$ encodes the same chronologies as $\varphi' = \varphi \wedge x - y \leq 5$, (i.e., $[\![\varphi]\!] = [\![\varphi']\!]$) because φ *implies* $x - y \leq 5$.

Expressiveness of the model. While the language of constraints we have just defined might seem very restrictive, we claim that it allows one to define most of the relevant constraints in archaeology, as defined at the beginning of this section:

Terminus post quem. A *Terminus post quem* is defined as a lower bound B on a given start or end date. Such constraints can be expressed by $\mathsf{beg}(p) \geq B$ and $\mathsf{end}(p) \geq B$, respectively.

Terminus ante quem. Symmetrically, a *Terminus ante quem* is defined as an upper bound B on a given start or end date. Such constraints correspond to $\mathsf{beg}(p) \leq B$ and $\mathsf{end}(p) \leq B$, respectively.

Date range. Ranges on dates are the conjunction of a *terminus post* and *ante quem*. In the example of Figure 1, the constraint on the start of K_1 is expressed as: $\mathsf{beg}(K_1) \geq 1200 \wedge \mathsf{beg}(K_1) \leq 1300$.

Duration constraints. Since the duration of a period p can be computed as $\mathsf{end}(p) - \mathsf{beg}(p)$, constraints (lower bounds, upper bound or ranges) on the duration of a period also fit our model. In the example of Figure 1, the range on the duration of K_1 is expressed as: $\mathsf{end}(K_1) - \mathsf{beg}(K_1) \geq 0 \wedge \mathsf{end}(K_1) - \mathsf{beg}(K_1) \leq 15$.

Sequence. A sequence of periods p and q means that q follows immediately after p. Thus, the end date of p is the start date of q, which is formalised as: $\mathsf{end}(p) - \mathsf{beg}(q) = 0$.

'Contemporaneity' synchronism. Periods p and q are contemporary, i.e. there is a non-empty intersection between the intervals $I_p = [\mathsf{beg}(p), \mathsf{end}(p)]$ and $I_q = [\mathsf{beg}(q), \mathsf{end}(q)]$. To understand how to model this, we consider the opposite statement: the intersection between I_p and I_q is empty iff either I_p follows strictly I_q or I_q follows strictly I_p. That is, $I_p \cap I_q = \emptyset$ iff $\mathsf{end}(p) < \mathsf{beg}(q) \vee \mathsf{end}(q) < \mathsf{beg}(p)$. This can be expressed by our constraints by taking the negation: $I_p \cap I_q \neq \emptyset$ iff $\mathsf{end}(p) \geq \mathsf{beg}(q) \wedge \mathsf{end}(q) \geq \mathsf{beg}(p)$.

'Starts during' synchronism. Period p starts during period q, i.e. the start of p is included in the interval $[\mathsf{beg}(q), \mathsf{end}(q)]$. This is formalised as: $\mathsf{beg}(p) \geq \mathsf{beg}(q) \wedge \mathsf{beg}(p) \leq \mathsf{end}(q)$.

'Ends during' synchronism. Period p ends during period q, i.e. the end of p is included in the interval $[\mathsf{beg}(q), \mathsf{end}(q)]$. This is formalised as: $\mathsf{end}(p) \geq \mathsf{beg}(q) \wedge \mathsf{end}(p) \leq \mathsf{end}(q)$.

'Inclusion'. Period p is included in period q, which can be formalised in our model as: $\mathsf{beg}(p) - \mathsf{beg}(q) \geq 0 \wedge \mathsf{end}(p) - \mathsf{end}(q) \leq 0$.

In addition to these constraints that come from the archaeological data, we *assume* from now on that all our constraints *imply* that all periods must start before they end. This can be achieved by taking the conjunction of any constraint with: $\bigwedge_{p \in \mathcal{P}} \mathsf{beg}(p) \leq \mathsf{end}(p)$.

Observe that one type of requirement that our constraints cannot express is *non-contemporaneity*, i.e. that the intersection between two periods p and q is empty. Indeed, non-contemporaneity means that *either* the end of p occurs strictly before the beginning of q *or* end of q occurs strictly before the beginning of p. However, our constraint language does not allow one to express disjunction.

2.3 Normalisation of constraints

In order to make our subsequent discussions easier, we will, from now on, consider a *normal form* for constraints, where constraints contain atomic constraints of the form $x - y \leq c$ only. Let φ be a constraint. We obtain $\mathsf{Norm}\,(\varphi)$, the normal form of φ by applying the following steps:

1. First, we make sure that there exists at least one atomic constraint for each pair of variables x and y, by taking the conjunction of φ with:

$$\bigwedge_{x \in \mathcal{V}(\mathcal{P})} \big(x - x \leq 0 \wedge x - z_0 \leq +\infty \wedge z_0 - x \leq 0\big) \wedge \bigwedge_{x,y \in \mathcal{V}(\mathcal{P}) \setminus \{z_0\}} x - y \leq +\infty\,.$$

   It is easy to check that the resulting constraint accepts the same set of chronologies than φ. Indeed, since we assume that z_0 is always null: $x - x \leq 0$ is equivalent to $x \leq x$; $x - z_0 \leq +\infty \wedge z_0 - x \leq 0$ is equivalent to $0 \leq x \leq +\infty$; and $x - y \leq +\infty$ should hold for all x, y since they take finite values.

2. We turn all atomic constraints into constraints of the form $x - y \leq c$. That is, we replace all atomic constraints $x \sim c$ by $x - z_0 \sim c$; and all constraints $x - y \geq c$ by $y - x \leq -c$.

3. Finally, whenever there are two different atomic constraints of the form $x - y \leq c$ and $x - y \leq c'$ in the resulting constraint, we keep the strongest one, i.e. $x - y \leq c$ if $c < c'$ and $x - y \leq c'$ otherwise.

The resulting is a normalised constraint $\mathsf{Norm}\,(\varphi)$. Observe that a normalised constraint always contain *exactly* $(2n+1)^2$ *atomic constraints of the form* $x - y \leq c$, where n is the number of periods; one for each pair of variables x and y in $\mathcal{V}(\mathcal{P})$ – this is actually the point of normalising constraints, even if the normalisation step might introduce some trivial atomic constraints. *From now on, we assume that all constraints are normalised.* We abuse notations and denote normalised constraint by non-normalised ones, writing, for instance, $x - y \leq 1$ instead of $\mathsf{Norm}\,(x - y \leq 1)$ which has 9 conjuncts.

Observe that this particular class of linear constraints has been studied before by Shostak [17]. They also form a special case of zones [1], and the normalised version of the constraint correspond to the Difference Bound Matrix [6] encoding the corresponding zone.

► **Example 1.** Consider again the example in Figure 1. In order to keep our example legible, we will consider only stratum S_1, king K_1 and the 'starts during' relationship between them. The following normalised constraint expresses exactly all the information from Figure 1 about S_1 and K_1, assuming $\mathcal{X} = \{z_0, \mathsf{beg}(S_1), \mathsf{beg}(K_1), \mathsf{end}(S_1), \mathsf{end}(K_1)\}$. Observe that the five last lines (marked 'Norm.') are here for normalisation purpose only.

$$\begin{aligned}
&\mathsf{end}(S_1) - \mathsf{beg}(S_1) \leq 100 \wedge \mathsf{beg}(S_1) - \mathsf{end}(S_1) \leq -20 && \text{(Length } S_1\text{)}\\
\wedge\ &\mathsf{end}(S_1) - \mathsf{beg}(S_1) \leq 15 \wedge \mathsf{beg}(S_1) - \mathsf{end}(S_1) \leq 0 && \text{(Length } K_1\text{)}\\
\wedge\ &\mathsf{beg}(K_1) - z_0 \leq 1300 \wedge z_0 - \mathsf{beg}(K_1) \leq -1200 && \text{(Start } K_1\text{)}\\
\wedge\ &\mathsf{end}(K_1) - z_0 \leq 1300 \wedge z_0 - \mathsf{end}(K_1) \leq -1200 && \text{(End } K_1\text{)}\\
\wedge\ &\mathsf{beg}(S_1) - z_0 \leq +\infty \wedge \mathsf{beg}(S_1) - \mathsf{beg}(K_1) \leq +\infty \wedge \mathsf{beg}(S_1) - \mathsf{end}(K_1) \leq +\infty && \text{(Norm.)}\\
\wedge\ &\mathsf{beg}(K_1) - \mathsf{beg}(S_1) \leq +\infty \wedge \mathsf{beg}(K_1) - \mathsf{end}(S_1) \leq +\infty && \text{(Norm.)}\\
\wedge\ &\mathsf{end}(S_1) - z_0 \leq +\infty \wedge \mathsf{end}(S_1) - \mathsf{beg}(K_1) \leq +\infty \wedge \mathsf{end}(S_1) - \mathsf{end}(K_1) \leq +\infty && \text{(Norm.)}\\
\wedge\ &\mathsf{end}(K_1) - \mathsf{beg}(S_1) \leq +\infty \wedge \mathsf{end}(K_1) - \mathsf{end}(S_1) \leq +\infty && \text{(Norm.)}\\
\wedge\ &\bigwedge_{x\in\mathcal{X}} x - x \leq 0 \wedge z_0 - \mathsf{beg}(S_1) \leq 0 \wedge z_0 - \mathsf{end}(S_1) \leq 0 && \text{(Norm.)}
\end{aligned}$$

3 Algorithmic manipulation of constraints

In this section, we show how the constraints from Section 2 can be manipulated algorithmically to answer the questions we have highlighted in introduction. We start by defining four meaningful problems on constraints, then show how the constraints can be expressed by means of weighted directed graphs, and finally give polynomial-time algorithms to solve those problems on the graphs. The main ideas of these techniques have been presented by Shostak [17], but our techniques for solving the sure- and possible- contemporaneity problems (that are motivated by the archaeological setting) are, as far as we know, original (and hence require dedicated proofs).

3.1 Four basic problems

Based on the motivations from the introduction, we focus on the four following problems.

Satisfiability. First, the *satisfiability* problem asks whether there is some chronology that satisfies a given constraint. If not, then the constraint contains a contradiction, for example: the constraint entails that some punctual period A should occur strictly before B, and, at the same time, that B should occur before A. Thus, the definition of this problem is as follows:

▶ **Problem 1.** *Given a constraint φ, the* satisfiability problem *asks whether $[\![\varphi]\!] \neq \emptyset$?*

If yes, we say that the constraint φ is *satisfiable*.

Tightening. Second, the *tightening* problem asks, given a constraint φ, to compute the tightest constraint φ' that represents the same set of chronologies. Intuitively, φ' represents the most precise information one can deduce from φ. Let us first define formally these notions.

Given two atomic constraints $\psi_1 = x - y \leq c_1$ and $\psi_2 = x - y \leq c_2$ (on the same variables x and y), we say that ψ_1 is *tighter* than ψ_2 (denoted $\psi_1 \preceq \psi_2$) iff $c_1 \leq c_2$. Intuitively, a constraint is *tighter* than another if it imposes a more stringent limitation on the possible values of the variables than the other (hence, the upper bound c_1 is smaller than or equal to c_2). Then, given two (normalised) constraints φ_1 and φ_2 on $\mathcal{P}$, we say that φ_1 is *tighter* than φ_2 (denoted $\varphi_1 \preceq \varphi_2$) iff each atomic constraint of φ_1 is tighter than the corresponding constraint in φ_2. Formally, $\varphi_1 \preceq \varphi_2$ iff for all x, y in $\mathcal{V}(\mathcal{P})$: $\psi_1 = x - y \leq c_1 \in \varphi_1$ and $\psi_2 = x - y \leq c_2 \in \varphi_2$ implies that $\psi_1 \preceq \psi_2$.

Observe that $\preceq$ is a partial order on constraints, which is not total. For instance, $x_1 - y_1 \leq 1 \wedge x_2 - y_2 \leq 2$ and $x_1 - y_1 \leq 2 \wedge x_2 - y_2 \leq 1$ are not comparable. Our definition of the order on constraints *implies* the intuition on the sets of chronologies they represent, i.e., for all constraints φ_1 and φ_2: $\varphi_1 \preceq \varphi_2$ implies $[\![\varphi_1]\!] \subseteq [\![\varphi_2]\!]$. However, the converse is not true. For instance, consider: $\varphi_1 = \mathsf{Norm}\,(x - y \leq 1 \wedge y - z \leq 1)$, and $\varphi_2 = \mathsf{Norm}\,(x - y \leq 1 \wedge y - z \leq 1 \wedge x - z \leq 2)$. It is easy to check that $[\![\varphi_1]\!] = [\![\varphi_2]\!]$ since the $x - z \leq 2$ atomic constraint of φ_2 is implied by its two other atomic constraints. Hence, in particular $[\![\varphi_1]\!] \subseteq [\![\varphi_2]\!]$, but, clearly $\varphi_1 \not\preceq \varphi_2$, because φ_1 constraints $x - z$ to be $\leq +\infty$, which is strictly weaker than $x - z \leq 2$. We can now define precisely the *tightening problem*:

▶ **Problem 2.** *Given a constraint φ, the* tightening problem *asks to compute the tightest (i.e., minimal wrt $\preceq$) constraint φ' s.t. $[\![\varphi']\!] = [\![\varphi]\!]$. Such a constraint φ' is called* tight.

Remark that this constraint φ' is necessarily unique. Indeed, assume it is not the case, and there are two constraints φ'_1 and φ'_2 s.t. $[\![\varphi'_1]\!] = [\![\varphi'_2]\!] = [\![\varphi]\!]$; $\varphi'_1 \preceq \varphi$; $\varphi'_2 \preceq \varphi$, but φ'_1 and φ'_2 are not comparable by $\preceq$ (that is, neither is tighter than the other). Then, we can consider instead the constraint ξ computed as follows: for each pair of variables x and y, we have in ξ the atomic constraint $x - y \leq \min(c_1, c_2)$, where c_1 and c_2 are the constants occurring in the atomic constraints on $x - y$ in φ'_1 and φ'_2 respectively (i.e., $x - y \leq c_1 \in \varphi'_1$ and $x - y \leq c_2 \in \varphi'_2$). By definition $\xi \preceq \varphi'_1$ and $\xi \preceq \varphi'_2$. Hence, $\xi \preceq \varphi$. Moreover, since $[\![\varphi'_1]\!] = [\![\varphi'_2]\!]$, we also have $[\![\xi]\!] = [\![\varphi'_1]\!] = [\![\varphi'_2]\!]$, hence $[\![\xi]\!] = [\![\varphi]\!]$. Thus, ξ is an even tighter constraint that can be used instead of φ'_1 and φ'_2.

Sure-contemporaneity. Third, the *sure-contemporaneity problem* asks whether two given periods p_1 and p_2 do *certainly* have a non-empty intersection given a constraint φ:

▶ **Problem 3.** *The* Sure-Contemporaneity Problem *asks, given a constraint φ (on set of periods $\mathcal{P}$) and two periods p_1 and p_2 in $\mathcal{P}$, whether φ* guarantees *that p_1 and p_2 intersect, i.e., whether $\mathcal{C}(p_1) \cap \mathcal{C}(p_2) \neq \emptyset$ for all $\mathcal{C} \in [\![\varphi]\!]$*

Possible-contemporaneity. Fourth, the *Possible-Contemporaneity problem* asks whether two given periods p_1 and p_2 can *possibly* have a non-empty intersection given a constraint φ:

▶ **Problem 4.** *The* possible-contemporaneity problem *asks whether a given constraint φ* does not exclude *a contemporaneity between two given periods p_1 and p_2, i.e. whether there is $\mathcal{C} \in [\![\varphi]\!]$ s.t. $\mathcal{C}(p_1) \cap \mathcal{C}(p_2) \neq \emptyset$.*

3.2 Graph-based algorithms

Let us now present polynomial time algorithms for solving the four problems highlighted in the previous section. The core of our approach consists, following Shostak [17], in translating each constraint φ into a directed weighted graph G_φ. Roughly speaking, the set of nodes in G_φ is the set of variables of φ, and each constraint $x - y \leq c$ is translated by a directed edge[6] from x to y, labelled by c.

More precisely, in our setting, a graph $G = \langle V, E, w\rangle$ is made up of a finite set of vertices V, a finite set of (directed) edges $E \subseteq V \times V$ and a weight function $w : E \to \mathbb{Z}$. Given a constraint φ on a set of periods $\mathcal{P}$, we build the graph $G_\varphi = \langle V, E, w\rangle$ as follows:

[6] Observe that this definition is consistent with the classical definitions in the literature on timed automata [6], but that the edges are reversed wrt the definitions generally used in the literature on constraint graphs [14].

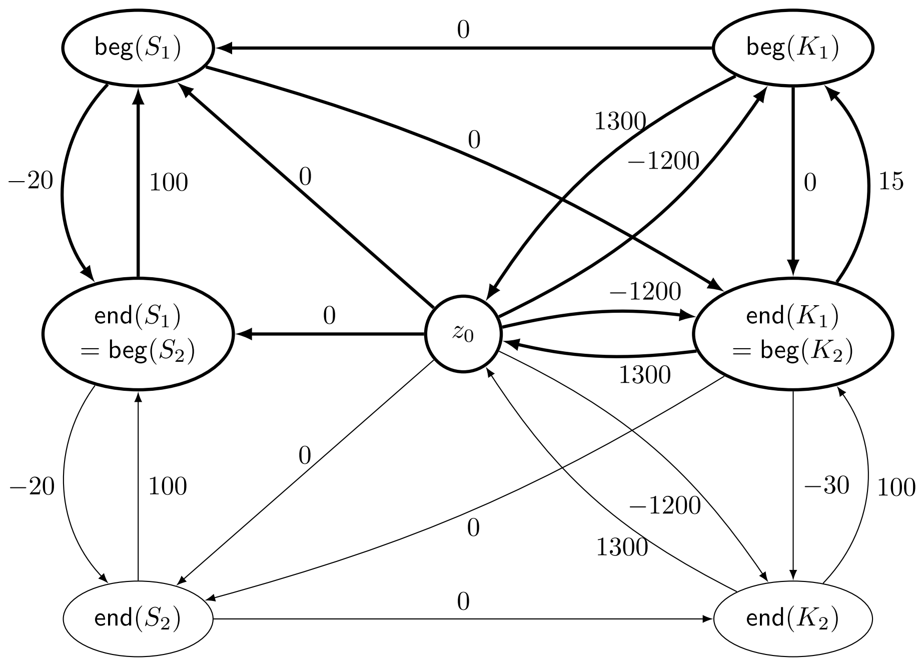

Figure 3 The graph Chrono for the constraint in Figure 1. To simplify presentation, nodes end(K_1) and beg(K_2) (respectively end(S_1) and beg(S_2)) have been merged and 0-labelled self-loops on the all nodes are not displayed. The bold part of the graph corresponds to the constraints on S_1 and K_1 only (see Example 1).

- $V = \mathcal{V}(\mathcal{P})$, i.e. there is a vertex for each variable in the constraint;
- $E = \{(x, y) \mid x - y \leq c \in \varphi$ and $c \neq +\infty\}$; and
- for all $(x, y) \in E$: $w(x, y) = c$ iff $x - y \leq c \in \varphi$. That is, there is an edge from x to y, labelled by c every time φ contains a non-trivial atomic constraint $x - y \leq c$ (by 'non-trivial', we mean that c is not $+\infty$). Thus, the edges encode exactly the set of φ's atomic constraints.

▶ **Example 2.** The graph corresponding to the full constraint modelling Chronoland (Figure 1) is given in Figure 3. The bold part of the graph corresponds to the constraint given in Example 1 (ranging on S_1 and K_1 only).

Now, we introduce our algorithms for our four problems given above.

Satisfiability checking and tightening. We address these two problems together as satisfiability can clearly be checked from the tightening of the constraint: clearly, φ is satisfiable iff the tightening of φ yields a constraint ξ s.t. $[\![\xi]\!] \neq \emptyset$. As said before, we rely on previous works for satisfiability and tightening. We start by recalling classical notions on graphs. Given a directed weighted graph $G = \langle V, E, w \rangle$, a *path* (from v_1 to v_k) is a finite sequence $\pi = v_1, v_2, \ldots, v_k$ of vertices ($v_i \in V$ for all i) s.t. $(v_i, v_{i+1}) \in E$ for all $1 \leq i \leq k - 1$. A *cycle* is a path $v_1, v_2, \ldots, v_k$ s.t. $v_1 = v_k$. The *weight* $w(\pi)$ of a path $\pi = v_1, v_2, \ldots, v_k$ is the sum of its edge weights, i.e. $\sum_{i=1}^{k-1} w(v_i, v_{i+1})$. A path π (and, in particular, a cycle) is *negative* iff $w(\pi) < 0$. A path $\pi = v_1, v_2, \ldots, v_k$ is called a *shortest path* from v_1 to v_k iff there is no other path π' from v_1 to v_k s.t. $w(\pi') < w(\pi)$ (observe that there could be several shortest paths from v_1 to v_k, but all of them have necessarily the same weight). In a graph that contains no negative cycle, it is well-known that there exists always a shortest path between two pairs of nodes, provided that there exists a path between those nodes. In such graphs G, we note $sp_G(x, y)$ the weight $w(\pi)$ of any shortest path π from x to y (and we let $sp_G(x, y) = +\infty$ if there is no path from x to y in G). The problem asking to compute $sp_G(x, y)$ for all pairs of nodes (x, y) (or to declare the value $sp_G(x, y)$ as undefined when the graph contains a negative cycle) is known in the literature as the *all-pairs shortest path problem* [12]. This problem allows us to solve the satisfiability and tightening problems. In [17], the author shows that:

▶ **Theorem 3** ([17]). *A constraint φ is satisfiable iff its graph G_φ contains no* negative cycle*.*

Moreover, the tightening of constraints has been considered in the setting of timed systems, and has been shown equivalent to the computation of all-pairs shortest paths:

▶ **Theorem 4** ([6]). *Given a* satisfiable *constraint φ (on set of variables $\mathcal{V}$ and with corresponding graph G_φ), the tightest constraint φ' s.t. $[\![\varphi]\!] = [\![\varphi']\!]$ is the constraint:*

$$\bigwedge_{x,y\in\mathcal{V}} x - y \leq sp_{G_\varphi}(x, y).$$

The reduction to shortest paths applies only when the constraint is satisfiable. Otherwise, the graph contains a negative cycle (by Theorem 3) and the notion of shortest path makes no sense.

Thus, in order to solve both satisfiability and tightening in practice, one can rely on one of the algorithms for the all-pairs shortest path problem from the literature (see [12] for a survey). All of these algorithms run in polynomial time. For example, one could use Johnson's algorithm [10], which runs in time $O(|V|^2 \log(|V|) + |V||E|)$ and detects negative cycles before computing all-pairs shortest paths, if the graph contains no negative cycle.

▶ **Example 5.** Let us come back to the Chronoland example. The graph in Figure 3 contains no negative cycle, hence the overall constraint φ of Chronoland is satisfiable. We now come to tightening. The result of the all-pairs shortest path computation is given in appendix (Figure 4) for reference. The most relevant results of computing all-pairs shortest paths in the graph of Figure 3 are summarised in Figure 2. For instance, the upper bound of 1260 on $\mathsf{beg}(S_1)$ is obtained by considering the atomic constraint of the form $\mathsf{beg}(S_1) - z_0 \leq sp_G(\mathsf{beg}(s_1), z_0)$ in the tightened constraint. The value $sp_G(\mathsf{beg}(s_1), z_0)$ is obtained by considering the path $\mathsf{beg}(S_1), \mathsf{end}(S_1), \mathsf{beg}(S_2), \mathsf{end}(S_2), \mathsf{end}(K_2), z_0$ of total weight $-20 + -20 + 1300 = 1260$. Other values are obtained similarly.

Checking sure-contemporaneity. Let us now explain how sure-contemporaneity (Problem 3) can be checked against the graph G_φ (corresponding to constraint φ) *in constant time*, provided that φ is tight. We start by defining the Inclusion Checking problem, which will be useful to this end. This problem checks whether the set of chronologies $[\![\varphi_1]\!]$ represented by a constraint φ_1 is included into the set of chronologies $[\![\varphi_2]\!]$ represented by another constraint φ_2:

▶ **Problem 5.** *The* Inclusion *problem asks, given two constraints φ_1 and φ_2, whether $[\![\varphi_1]\!] \subseteq [\![\varphi_2]\!]$.*

We have already seen in an example above that $\varphi_1 \preceq \varphi_2$ implies $[\![\varphi_1]\!] \subseteq [\![\varphi_2]\!]$, but that, in general, the reverse implication is not true. It becomes, however, true, when the constraints have been tightened. As a matter of fact, having φ_1 tight is sufficient (see [3] for a reference):

▶ **Proposition 6.** *For all pairs of constraints φ_1 and φ_2, the two following statements hold:*

$$\varphi_1 \preceq \varphi_2 \textit{ implies } [\![\varphi_1]\!] \subseteq [\![\varphi_2]\!],$$
$$\big([\![\varphi_1]\!] \subseteq [\![\varphi_2]\!] \textit{ and } \varphi_1 \textit{ is tight}\big) \textit{ implies } \varphi_1 \preceq \varphi_2.$$

We now come back to the sure-contemporaneity problem, and show that it can be reduced to the inclusion problem. The definition of the sure-contemporaneity problem means that, for all chronologies $\mathcal{C} \in [\![\varphi]\!]$: p_1 and p_2 intersect in $\mathcal{C}$, i.e., $\ell(\mathcal{C}(p_1)) \leq u(\mathcal{C}(p_2))$ and $\ell(\mathcal{C}(p_2)) \leq u(\mathcal{C}(p_1))$. Clearly, this holds iff $[\![\varphi]\!] \subseteq [\![\varphi^{\text{sync}}_{p_1,p_2}]\!]$, where:

$$\varphi^{\text{sync}}_{p_1,p_2} = \mathsf{beg}(p_1) \leq \mathsf{end}(p_2) \wedge \mathsf{beg}(p_2) \leq \mathsf{end}(p_1). \tag{1}$$

However, $\varphi^{\text{sync}}_{p_1,p_2}$ is equivalent to $\mathsf{beg}(p_1) - \mathsf{end}(p_2) \leq 0 \wedge \mathsf{beg}(p_2) - \mathsf{end}(p_1) \leq 0$. Thus, using Proposition 6, and assuming φ is tight, we deduce that $[\![\varphi]\!] \subseteq [\![\varphi^{\text{sync}}_{p_1,p_2}]\!]$ iff φ constraints $\mathsf{beg}(p_1) - \mathsf{end}(p_2)$ and $\mathsf{beg}(p_2) - \mathsf{end}(p_1)$ to be non-positive. Thus, we obtain a constant time procedure to check the sure-contemporaneity of two periods p_1 and p_2 on the graph G_φ of a tight constraint φ:

▶ **Proposition 7.** *For all tight constraint φ on $\mathcal{P}$ (with corresponding graph $G_\varphi = (V, E, w)$), for all pairs of periods p_1 and p_2, there is a sure-contemporaneity between p_1 and p_2 in φ iff $w(\mathsf{beg}(p_1), \mathsf{end}(p_2)) \leq 0$ and $w(\mathsf{beg}(p_2), \mathsf{end}(p_1)) \leq 0$ (assuming $w(x, y) = +\infty$ when $(x, y) \notin E$).*

▶ **Example 8.** To answer our question 'has K_1 built S_2?' from the introduction, we can check whether there is a sure-contemporaneity between K_1 and S_2. By proposition 7, there is *no sure-contemporaneity* iff either $sp_{\mathsf{Chrono}}(\mathsf{beg}(K_1), \mathsf{end}(S_2)) > 0$ or $sp_{\mathsf{Chrono}}(\mathsf{beg}(S_2), \mathsf{end}(K_1)) > 0$ (see Figure 3 for Chrono). While the path $\mathsf{beg}(K_1), \mathsf{beg}(S_1), \mathsf{end}(S_1), \mathsf{beg}(S_2), \mathsf{end}(S_2)$ is indeed negative, one can check that there all paths from $\mathsf{beg}(S_2)$ to $\mathsf{end}(K_1)$ are positive (actually, $sp_{\mathsf{Chrono}}(\mathsf{beg}(S_2), \mathsf{end}(K_1)) = 80$, see Appendix A). Hence, there is no sure-contemporaneity between K_1 and S_2, so the available archaeological data does not allow to say for sure that K_1 built S_2.

Checking Possible-Contemporaneity. As with the sure-contemporaneity problem, we first rephrase the definition of the possible-contemporaneity problem using the $\varphi^{\text{sync}}_{p_1,p_2}$ constraint from equation (1). Since there must be only one chronology compatible with the constraint φ in which p_1 and p_2 intersect, and since the set of such chronologies is characterised by $\varphi^{\text{sync}}_{p_1,p_2}$ from equation (1), we have:

▶ **Lemma 9.** *For all constraints φ on set of periods $\mathcal{P}$, and all pairs of periods $p_1, p_2 \in \mathcal{P}$: there is a possible-contemporaneity between p_1 and p_2 iff $\varphi \wedge \varphi^{sync}_{p_1,p_2}$ is satisfiable.*

From this characterisation, we obtain a simple algorithm to check possible-contemporaneity on tight constraints (in the spirit of Proposition 7 for sure-contemporaneity):

▶ **Proposition 10.** *For all tight constraints φ on $\mathcal{P}$ (with corresponding graph $G_\varphi = (V, E, w)$), for all pairs of periods p_1 and p_2, there is a possible-contemporaneity between p_1 and p_2 in φ iff $w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) \geq 0$ and $w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) \geq 0$ (assuming $w(x, y) = +\infty$ when $(x, y) \notin E$).*

▶ **Example 11.** We come back again to the Chronoland example and our question asking whether K_1 has built S_2. We check whether there is a possible-contemporaneity between K_1 and S_2 (remember from Example 8 that there is no *sure*-contemporaneity between them).

By Proposition 10, there is *no* possible-contemporaneity iff $sp_{\mathsf{Chrono}}(\mathsf{end}(S_2), \mathsf{beg}(K_1)) < 0$ or $sp_{\mathsf{Chrono}}(\mathsf{end}(K_1), \mathsf{beg}(S_2)) < 0$. The latter holds since the path $\mathsf{end}(K_1), \mathsf{beg}(K_1), \mathsf{beg}(S_1)$, $\mathsf{end}(S_1), \mathsf{beg}(S_2)$ has weight $15 + -20 = -5$ (see Appendix A). Hence, there is *no possible-contemporaneity* between K_1 and S_2. This result is stronger than the one from Example 8 and allows one to rule out for sure that K_1 built stratum S_2.

The result of Proposition 10 is not straightforward and requires a proof. We start by giving some intuitions. First observe that we consider a constraint φ, which we assume to be satisfiable (otherwise there is trivially no possible-contemporaneity), and tight. Then, the proof is based on the following observation: taking the conjunction of φ and $\varphi^{sync}_{p_1,p_2}$ (Lemma 9) amounts to computing the graph $G' = (V, E', w')$ from $G_\varphi = (V, E, w)$ as follows (assuming $w(x, y) = +\infty$ if $(x, y) \notin E$):

- $E' = E \cup \big\{\big(\mathsf{beg}(p_2), \mathsf{end}(p_1)\big), \big(\mathsf{beg}(p_1), \mathsf{end}(p_2)\big)\big\}$;
- $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) = \min\{w(\mathsf{beg}(p_2), \mathsf{end}(p_1)), 0\}$;
- $w'(\mathsf{beg}(p_1), \mathsf{end}(p_2)) = \min\{w(\mathsf{beg}(p_1), \mathsf{end}(p_2)), 0\}$;
- $w'(e) = w(e)$ for all $e \in E\big\{\big(\mathsf{beg}(p_2), \mathsf{end}(p_1)\big), \big(\mathsf{beg}(p_1), \mathsf{end}(p_2)\big)\big\}$.

Thus, G' is obtained from G_φ by setting the weights of $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$ and $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$ to 0 if they had non-negative weights in G_φ. One can check that the constraint corresponding to G' is equivalent to $\varphi \wedge \varphi^{\mathrm{sync}}_{p_1,p_2}$, so there is a possible-contemporaneity between p_1 and p_2 iff there is no negative cycle in G', by Lemma 9.

Now assume that $\varphi \wedge \varphi^{\mathrm{sync}}_{p_1,p_2}$ is *not satisfiable* and thus G' contains a negative cycle. Since φ is tight and satisfiable, G_φ contains no negative cycle, hence the negative cycle in G' comes necessarily from the modification we have performed on $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$ and $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$. Thus, there is, in G', at least one negative cycle that contains $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$ or $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$. For the sake of the discussion, let us assume the negative cycle contains $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$ and not $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$, hence, the only reason for this negative cycle to exist in G' is because we have set $w(\mathsf{beg}(p_2), \mathsf{end}(p_1))$ to 0, and thus, there is a negative *path* from $\mathsf{end}(p_1)$ to $\mathsf{beg}(p_2)$ in G' and G_φ. Since φ is tight, this implies that $(\mathsf{end}(p_1), \mathsf{beg}(p_2))$ exists and has negative weight in G_φ. Conversely, if $w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) < 0$, then, there is necessarily a negative cycle containing $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$ in G', since $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) \leq 0$, by construction. This explains intuitively why checking that both $w(\mathsf{end}(p_2), \mathsf{beg}(p_1))$ and $w(\mathsf{end}(p_1), \mathsf{beg}(p_2))$ are non-negative is necessary and sufficient to check possible-contemporaneity. This can be performed in $O(1)$ on *tight* (and satisfiable) constraints. A formal proof of the proposition is given in Appendix B.

4 Conclusion and future works

This paper presents a theoretical framework for the modelling of chronological constraints relevant to archaeological research. Within this framework, algorithms have been presented to solve four basic chronological problems (satisfiability, tightening, sure- and possible-contemporaneity) that are usually addressed in a non-formal way by the archaeological community. As shown here for a toy example featuring only two kings and two archaeological strata (Figure 1), solving these problems manually is tedious and error-prone, especially when it comes to obtaining optimal bounds on dates and duration (tightening). On real-life archaeological cases, featuring dozens of periods and synchronisms, obtaining reliable and optimal results is virtually impossible without the help of a formal computational approach as the one advocated for here.

We further contend that the application of our algorithms to real-life archaeological test-cases might provide significant advances to current chronological debates, through the detection of (yet unnoticed) unsatisfiable sets of chronological constraints, and through the computation of improved chronological estimates through our tightening procedure. An automated technique also allows archaeologist to test quickly the implications of new hypotheses on chronological models.

The next steps of this research are therefore both practical and theoretical. On the *practical* side, we wish to develop a comprehensive methodology for chronological research in archaeology, including a user-friendly tool that will allow archaeologists to specify and manipulate their own chronological models, through a dedicated high-level constraint language. We also want to apply this methodology to concrete case studies from the archaeological literature. On the *theoretical* side, it is clear that the development of concrete case studies will raise new theoretical questions. One such questions we can already mention is 'how to provide the archaeological user with a meaningful *witness* of non-satisfiability?' when a constraint is found to be unsatisfiable. In such as case, the user would wish to understand why the system is unsatisfiable, and also be provided with hints as to how the system could be rendered satisfiable through the removal (or relaxing) of a limited number of constraints.

Acknowledgements. Eythan Levy was supported by the Center for Absorption in Science, the Ministry of Absorption, the State of Israel.

References

1 Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994. `doi:10.1016/0304-3975(94)90010-8`.

2 Gerd Behrmann, Alexandre David, and Kim G. Larsen. A tutorial on Uppaal. In *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM-RT 2004*, volume 3185 of *Lecture Notes in Computer Science*, pages 200–236. Springer, September 2004.

3 Johan Bengtsson. *Clocks, DBM, and States in Timed Systems.* PhD thesis, Uppsala University, 2002.

4 Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In *Lectures on Concurrency and Petri Nets, Advances in Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 87–124. Springer, 2003. `doi:10.1007/978-3-540-27755-2_3`.

5 Patricia Bouyer, Maximilien Colange, and Nicolas Markey. Symbolic optimal reachability in weighted timed automata. In *CAV'16*, volume 9779 of *LNCS*, pages 513–530. Springer, 2016. `doi:10.1007/978-3-319-41528-4`.

6 David L. Dill. Timing assumptions and verification of finite-state concurrent systems. In *Automatic Verification Methods for Finite State Systems, International Workshop, Grenoble, France, June 12-14, 1989, Proceedings*, volume 407 of *Lecture Notes in Computer Science*, pages 197–212. Springer Verlag, 1989. `doi:10.1007/3-540-52148-8_17`.

7 E. C. Harris. *Principles of Archaeological Stratigraphy.* Academic Press, New York. NY, 1979.

8 Felix Höflmayer. The date of the Minoan Santorini eruption: Quantifying the "offset". *Radiocarbon*, 54:435–448, 2012.

9 Mads Kähler Holst. Complicated relations and blind dating: Formal analysis of relative chronological structures. In Caitlin E. Buck and Andrew R. Millard, editors, *Tools for Constructing Chronologies*, pages 129–147. Springer, London, 2004.

10 Donald B. Johnson. Efficient algorithms for shortest paths in sparse networks. *J. ACM*, 24(1):1–13, 1977. `doi:10.1145/321992.321993`.

11 Alfred H. Kromholz. Business and industry in archaeology. In Paul Ahström, editor, *High, Middle or Low? (Part 1)*, pages 119–137. Paul Ahströms Förlag, Gothenburg, 1987.

12 A. Madkour, W. G. Aref, F. U. Rehman, M. Abdur Rahman, and S. Basalamah. A Survey of Shortest-Path Algorithms. Technical Report CoRR abs/1705.02044, Cornell University Library, arXiv.org, 2017. URL: `https://arxiv.org/abs/1705.02044`.

13 Amihai Mazar. The debate over the chronology of the Iron Age in the southern Levant. In *The Bible and Radiocarbon Dating, Archaeology, Text and Science*, pages 15–30, 2005.

14 Itay Meiri. Combining qualitative and quantitative constraints in temporal reasoning. *Artificial Intelligence*, 87(1):343–385, 1996. `doi:10.1016/0004-3702(95)00109-3`.

15 Christopher Bronk Ramsey. Radiocarbon calibration and analysis of stratigraphy: the OxCal program. *Radiocarbon*, 37(2):425–430, 1995.

16 Ilan Sharon. Partial order scalogram analysis of relations – a mathematical approach to the analysis of stratigraphy. *Journal of Archaeological Science*, 22:751–767, 1995.

17 Robert E. Shostak. Deciding linear inequalities by computing loop residues. *J. ACM*, 28(4):769–779, 1981. `doi:10.1145/322276.322288`.

A Shortest paths in the Chronoland example

The all-pairs shortest path matrix of the Chronoland example (graph in Figure 3) is given in Figure 4. The entry in row i column j gives the weight of the shortest path from i to j.

B Proof of Proposition 10

Statement of Proposition 10: *For all tight constraints φ on $\mathcal{P}$ (with corresponding graph $G_\varphi = (V, E, w)$), for all pairs of periods p_1 and p_2, there is a possible-contemporaneity between p_1 and p_2 in φ iff $w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) \geq 0$ and $w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) \geq 0$ (assuming $w(x, y) = +\infty$ when $(x, y) \notin E$).*

Proof. We prove both directions of the iff. First let us show that a possible-contemporaneity between p_1 and p_2 implies $w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) \geq 0$ and $w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) \geq 0$. We prove the contraposition, i.e., if either $w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) < 0$ or $w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) < 0$, then there is no possible-contemporaneity between p_1 and p_2.

Assume that $w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) < 0$ (the case $w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) < 0$ is symmetrical), and consider the graph $G' = \langle V, E', w' \rangle$ obtained, as described above. By definition of G',

$$
\begin{array}{c|ccccccccc}
 & z_0 & \mathsf{beg}(S_1) & \mathsf{end}(S_1) & \mathsf{beg}(S_2) & \mathsf{end}(S_2) & \mathsf{beg}(K_1) & \mathsf{end}(K_1) & \mathsf{beg}(K_2) & \mathsf{end}(K_2) \\
\hline
z_0 & 0 & -1,200 & -1,220 & -1,220 & -1,240 & -1,200 & -1,200 & -1,200 & -1,240 \\
\mathsf{beg}(S_1) & 1,260 & 0 & -20 & -20 & -40 & 15 & 0 & 0 & -40 \\
\mathsf{end}(S_1) & 1,280 & 80 & 0 & 0 & -20 & 80 & 80 & 80 & -20 \\
\mathsf{beg}(S_2) & 1,280 & 80 & 0 & 0 & -20 & 80 & \mathbf{80} & 80 & -20 \\
\mathsf{end}(S_2) & 1,300 & 100 & 80 & 80 & 0 & \boxed{100} & 100 & 100 & 0 \\
\mathsf{beg}(K_1) & 1,260 & 0 & -20 & -20 & \mathbf{-40} & 0 & 0 & 0 & -40 \\
\mathsf{end}(K_1) & 1,270 & 15 & -5 & \boxed{-5} & -25 & 15 & 0 & 0 & -30 \\
\mathsf{beg}(K_2) & 1,270 & 15 & -5 & -5 & -25 & 15 & 0 & 0 & -30 \\
\mathsf{end}(K_2) & 1,300 & 100 & 80 & 80 & 60 & 100 & 100 & 100 & 0
\end{array}
$$

Figure 4 The all-pairs shortest paths for the Chronoland example. **Bold** numbers highlight the values referred to in Example 8 where sure-contemporaneity between K_1 and S_2 is checked (and found not to hold). $\boxed{\text{Boxed}}$ numbers highlight the values referred to in Example 11, where possible-contemporaneity between K_1 and S_2 is checked (and found not to hold either).

and since $w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) < 0$: $w'(\mathsf{end}(p_2), \mathsf{beg}(p_1)) = w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) < 0$. Let us show that the cycle

$$\mathsf{end}(p_2), \mathsf{beg}(p_1), \mathsf{end}(p_1), \mathsf{beg}(p_2), \mathsf{end}(p_2)$$

is negative in G'. Indeed, we have:

$$\begin{aligned}
&w'(\mathsf{beg}(p_1), \mathsf{end}(p_1)) \leq 0 && \text{True in all constraints: the beginning occurs before the end}\\
&w'(\mathsf{beg}(p_2), \mathsf{end}(p_2)) \leq 0 && \text{Same argument}\\
&w'(\mathsf{end}(p_1), \mathsf{beg}(p_2)) \leq 0 && \text{By construction of } G'\\
&w'(\mathsf{end}(p_2), \mathsf{beg}(p_1)) < 0 && \text{By the above arguments.}
\end{aligned}$$

Hence, the weight of the $\mathsf{end}(p_2), \mathsf{beg}(p_1), \mathsf{end}(p_1), \mathsf{beg}(p_2), \mathsf{end}(p_2)$ cycle is indeed negative. However, by construction G' is the graph that corresponds to $\varphi \wedge \varphi^{\text{sync}}_{p_1,p_2}$, hence this constraint is not satisfiable. By Lemma 9, there is no possible-contemporaneity between p_1 and p_2 in φ.

For the other direction, let us show that:

$$w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) \geq 0 \text{ and } w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) \geq 0 \tag{2}$$

implies that there is a possible-contemporaneity between p_1 and p_2 in φ. Following Lemma 9, we show that (2) implies that $\varphi \wedge \varphi^{\text{sync}}_{p_1,p_2}$ is satisfiable. To show this, we rely again on the graph G' described above: we proceed by contradiction and assume that (2) holds but that G' contains a negative cycle. We consider several cases:

1. The negative cycle contains neither the $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$, nor the $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$ edge. Since these two edges are the only ones that have been modified when building G' from G_φ, we conclude that the negative cycle is already present in G_φ. This is a contradiction since we have assumed that φ is satisfiable.
2. The negative cycle contains only edge among $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$ and $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$. Wlog, we assume that the negative cycle of G' contains $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$, i.e. it is of the form:

   $$\mathsf{beg}(p_2) \xrightarrow{w'(\mathsf{beg}(p_2),\mathsf{end}(p_1))} \underbrace{\mathsf{end}(p_1) \xrightarrow{w_1} v_1 \xrightarrow{w_2} \cdots \xrightarrow{w_n} v_n \xrightarrow{w_{n+1}} \mathsf{beg}(p_2)}_{\pi}$$

   where $w_1 = w'(\mathsf{end}(p_1), v_1)$, $w_i = w'(v_{i-1}, v_i)$ for all $2 \leq i \leq n$, $w_{n+1} = w'(v_n, \mathsf{beg}(p_2))$, and neither $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$, nor $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$ occur in π. Since this cycle is negative in G':

   $$w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) + \sum_{i=1}^{n+1} w_i < 0.$$

   Recall that, by construction of G': $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) \leq 0$. We consider two further sub-cases:
   a. If $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) = 0$, then $\sum_{i=1}^{n} w_i < 0$, i.e., the total weight of π is non-positive. However, since all edges occurring in π occur with the same weight in G_φ, and since π starts in $\mathsf{end}(p_1)$ and ends in $\mathsf{beg}(p_2)$, we conclude that the shortest path between $\mathsf{end}(p_1)$ and $\mathsf{beg}(p_2)$ is < 0 in G_φ. Since φ is assumed to be tight, this implies that $w(\mathsf{end}(p_1), \mathsf{beg}(p_2)) < 0$, which is a contradiction with our hypothesis (2).
   b. If $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) < 0$, then by construction of G', this edge had the same weight in G_φ, i.e., $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) = w(\mathsf{beg}(p_2), \mathsf{end}(p_1)) < 0$. Since all the edges in π were also present in G_φ with the same weight, we conclude that the negative cycle we have identified in G' is also present in G_φ. This is a contradiction since we have assumed that φ is satisfiable.

3. The negative cycle contains both $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$ and $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$, i.e. it is of the form:

$$\mathsf{beg}(p_1) \xrightarrow{w'(\mathsf{beg}(p_1),\mathsf{end}(p_2))} \underbrace{\mathsf{end}(p_2) \xrightarrow{w_1} v_1 \xrightarrow{w_2} \cdots \xrightarrow{w_n} v_n \xrightarrow{w_{n+1}} \mathsf{beg}(p_2)}_{\pi_1}$$

$$\xrightarrow{w'(\mathsf{beg}(p_2),\mathsf{end}(p_1))} \underbrace{\mathsf{end}(p_1) \xrightarrow{w'_1} v'_1 \xrightarrow{w'_2} \cdots \xrightarrow{w'_\ell} v'_\ell \xrightarrow{w'_{\ell+1}} \mathsf{beg}(p_1)}_{\pi_2}$$

where $w_1 = w'(\mathsf{end}(p_2), v_1)$; for all $2 \le i \le n$: $w_i = w'(v_{i-1}, v_i)$; $w_{n+1} = w'(v_n, \mathsf{beg}(p_2))$; $w'_1 = w'(\mathsf{end}(p_1), v'_1)$; for all $1 \le i \le \ell$: $w'_i = w'(v'_{i-1}, v'_i)$; $w'_{\ell+1} = w'(v'_\ell, \mathsf{beg}(p_1))$; and $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$, $(\mathsf{beg}(p_1), \mathsf{end}(p_2))$ occur neither in π_1 nor in π_2. Since this cycle is negative in G', we have:

$$w'(\mathsf{beg}(p_1), \mathsf{end}(p_2)) + \sum_{i=1}^{n+1} w_i + w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) + \sum_{i=1}^{\ell+1} w'_i < 0 \qquad (3)$$

Since, by construction of G': $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) \le 0$ and $w'(\mathsf{beg}(p_1), \mathsf{end}(p_2)) \le 0$, we consider four further sub-cases:

a. First, $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) = w'(\mathsf{beg}(p_1), \mathsf{end}(p_2)) = 0$. Then, (3) yields:

$$\sum_{i=1}^{n+1} w_i + \sum_{i=1}^{\ell+1} w'_i < 0.$$

Hence, one of these two sums must be < 0. Wlog, let us assume $\sum_{i=1}^{n+1} w_i < 0$, i.e., the weight of π_1 is non-positive (the arguments carry on when the overall weight of π_2 is non-positive instead). Since all the edges of π_1 were already present in G with the same weights, we conclude that the shortest path from $\mathsf{end}(p_2)$ to $\mathsf{beg}(p_2)$ is < 0 in G. Since we have assumed that φ is tight, this implies that $w(\mathsf{end}(p_2), \mathsf{beg}(p_2)) < 0$, hence, φ contains an atomic constraint of the form $\mathsf{end}(p_2) - \mathsf{beg}(p_2) \le c$ from some $c < 0$. However, this renders φ unsatisfiable, because all constraints imply that $\mathsf{beg}(p) - \mathsf{end}(p) \le 0$ for all periods p. Contradiction.

b. Second $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) < 0$ and $w'(\mathsf{beg}(p_1), \mathsf{end}(p_2)) = 0$. Then, (3) yields:

$$\sum_{i=1}^{n+1} w_i + w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) + \sum_{i=1}^{\ell+1} w'_i < 0. \qquad (4)$$

However, by construction of G', $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) < 0$ implies that the edge $(\mathsf{beg}(p_2), \mathsf{end}(p_1))$ was already in G with the same weight, i.e. $w(\mathsf{beg}(p_2), \mathsf{end}(p_1)) < 0$. Moreover, all edges in π_1 and π_2 are also present in G with the same weight as in G'. Thus, we conclude from (4) that the

$$\mathsf{end}(p_2), v_1, \ldots, v_n, \mathsf{beg}(p_2), \mathsf{end}(p_1), v'_1, \ldots, v'_\ell, \mathsf{beg}(p_1)$$

path exists in G with non-positive weight. Hence, the shortest path from $\mathsf{end}(p_2)$ to $\mathsf{beg}(p_1)$ in G has weight < 0. Since we have assumed that φ is tight, we conclude that $w(\mathsf{end}(p_2), \mathsf{beg}(p_1)) < 0$, which contradict our hypothesis (2). Contradiction.

c. Third, $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) = 0$ and $w'(\mathsf{beg}(p_1), \mathsf{end}(p_2)) < 0$ is treated as the previous case.

d. Finally, when $w'(\mathsf{beg}(p_2), \mathsf{end}(p_1)) < 0$ and $w'(\mathsf{beg}(p_1), \mathsf{end}(p_2)) < 0$, we conclude that all the edges in the negative cycle we have identified are already present in G with the same weight, i.e., the $\mathsf{beg}(p_1), \mathsf{end}(p_2), v_1, \ldots, v_n, \mathsf{beg}(p_2), \mathsf{end}(p_1), v'_1, \ldots, v'_\ell, \mathsf{beg}(p_1)$ cycle has negative weight in G, hence φ is not satisfiable. Contradiction. ◀