

Volume 25

Number 3

ACTA CYBERNETICA

Editor-in-Chief: Tibor Csendes (Hungary)

Managing Editor: Boglárka G.-Tóth (Hungary)

Assistant to the Managing Editor: Attila Tanács (Hungary)

Associate Editors:

Michał Baczyński (Poland)

Hans L. Bodlaender (The Netherlands)

Gabriela Csurka (France)

János Demetrovics (Hungary)

József Dombi (Hungary)

Rudolf Ferenc (Hungary)

Zoltán Fülöp (Hungary)

Zoltán Gingl (Hungary)

Tibor Gyimóthy (Hungary)

Zoltan Kato (Hungary)

Dragan Kukolj (Serbia)

László Lovász (Hungary)

Kálmán Palágyi (Hungary)

Dana Petcu (Romania)

Andreas Rauh (France)

Heiko Vogler (Germany)

Gerhard J. Woeginger (The Netherlands)

Szeged, 2022

ACTA CYBERNETICA

Information for authors. Acta Cybernetica publishes only original papers in the field of Computer Science. Manuscripts must be written in good English. Contributions are accepted for review with the understanding that the same work has not been published elsewhere. Papers previously published in conference proceedings, digests, preprints are eligible for consideration provided that the author informs the Editor at the time of submission and that the papers have undergone substantial revision. If authors have used their own previously published material as a basis for a new submission, they are required to cite the previous work(s) and very clearly indicate how the new submission offers substantively novel or different contributions beyond those of the previously published work(s). There are no page charges. An electronic version of the published paper is provided for the authors in PDF format.

Manuscript Formatting Requirements. All submissions must include a title page with the following elements: title of the paper; author name(s) and affiliation; name, address and email of the corresponding author; an abstract clearly stating the nature and significance of the paper. Abstracts must not include mathematical expressions or bibliographic references.

References should appear in a separate bibliography at the end of the paper, with items in alphabetical order referred to by numerals in square brackets. Please prepare your submission as one single PostScript or PDF file including all elements of the manuscript (title page, main text, illustrations, bibliography, etc.).

When your paper is accepted for publication, you will be asked to upload the complete electronic version of your manuscript. For technical reasons we can only accept files in LaTeX format. It is advisable to prepare the manuscript following the guidelines described in the author kit available at <https://cyber.bibl.u-szeged.hu/index.php/actcybern/about/submissions> even at an early stage.

Submission and Review. Manuscripts must be submitted online using the editorial management system at <https://cyber.bibl.u-szeged.hu/index.php/actcybern/submission/wizard>. Each submission is peer-reviewed by at least two referees. The length of the review process depends on many factors such as the availability of an Editor and the time it takes to locate qualified reviewers. Usually, a review process takes 6 months to be completed.

Subscription Information. Acta Cybernetica is published by the Institute of Informatics, University of Szeged, Hungary. Each volume consists of four issues, two issues are published in a calendar year. Subscription rates for one issue are as follows: 5000 Ft within Hungary, €40 outside Hungary. Special rates for distributors and bulk orders are available upon request from the publisher. Printed issues are delivered by surface mail in Europe, and by air mail to overseas countries. Claims for missing issues are accepted within six months from the publication date. Please address all requests to:

Acta Cybernetica, Institute of Informatics, University of Szeged
P.O. Box 652, H-6701 Szeged, Hungary
Tel: +36 62 546 396, Fax: +36 62 546 397, Email: acta@inf.u-szeged.hu

Web access. The above information along with the contents of past and current issues are available at the Acta Cybernetica homepage <https://cyber.bibl.u-szeged.hu/>.

EDITORIAL BOARD

Editor-in-Chief:

Tibor Csendes
Department of Computational Optimization
University of Szeged, Hungary
csendes@inf.u-szeged.hu

Managing Editor:

Boglárka G.-Tóth
Department of Computational Optimization
University of Szeged, Hungary
boglarka@inf.u-szeged.hu

Assistant to the Managing Editor:

Attila Tanács
Department of Image Processing
and Computer Graphics
University of Szeged, Hungary
tanacs@inf.u-szeged.hu

Associate Editors:

Michał Baczyński
Faculty of Science and Technology,
University of Silesia in Katowice,
Poland
michal.baczynski@us.edu.pl

Hans L. Bodlaender
Institute of Information and
Computing Sciences, Utrecht
University, The Netherlands
h.l.bodlaender@uu.nl

Gabriela Csurka
Naver Labs, Meylan, France
gabriela.csurka@naverlabs.com

János Demetrovics
MTA SZTAKI, Budapest, Hungary
demetrovics@sztaki.hu

József Dombi
Department of Computer Algorithms
and Artificial Intelligence, University of
Szeged, Hungary
dombi@inf.u-szeged.hu

Rudolf Ferenc
Department of Software Engineering,
University of Szeged, Hungary
ferenc@inf.u-szeged.hu

Zoltán Fülöp
Department of Foundations of
Computer Science, University of
Szeged, Hungary
fulop@inf.u-szeged.hu

Zoltán Gingl
Department of Technical Informatics,
University of Szeged, Hungary
gingl@inf.u-szeged.hu

Tibor Gyimóthy

Department of Software Engineering,
University of Szeged, Hungary
gyimothy@inf.u-szeged.hu

Zoltan Kato

Department of Image Processing and
Computer Graphics, University of
Szeged, Hungary
kato@inf.u-szeged.hu

Dragan Kukolj

RT-RK Institute of Computer Based
Systems, Novi Sad, Serbia
dragan.kukolj@rt-rk.com

László Lovász

Department of Computer Science,
Eötvös Loránd University, Budapest,
Hungary
lovasz@cs.elte.hu

Kálmán Palágyi

Department of Image Processing and
Computer Graphics, University of
Szeged, Hungary
palagyi@inf.u-szeged.hu

Dana Petcu

Department of Computer Science, West
University of Timisoara, Romania
petcu@info.uvt.ro

Andreas Rauh

School II – Department of Computing
Science, Group Distributed Control in
Interconnected Systems, Carl von
Ossietzky Universität Oldenburg,
Germany
andreas.rauh@uni-oldenburg.de

Heiko Vogler

Department of Computer Science,
Dresden University of Technology,
Germany
Heiko.Vogler@tu-dresden.de

Gerhard J. Woeginger

Department of Mathematics and
Computer Science, Eindhoven
University of Technology, The
Netherlands

Domain Semirings United

Uli Fahrenberg^a, Christian Johansen^b, Georg Struth^c,
and Krzysztof Ziemiański^d

Abstract

Domain operations on semirings have been axiomatised in two different ways: by a map from an additively idempotent semiring into a boolean subalgebra of the semiring bounded by the additive and multiplicative unit of the semiring, or by an endofunction on a semiring that induces a distributive lattice bounded by the two units as its image. This note presents classes of semirings where these approaches coincide.

Keywords: semirings, quantales, domain operations

1 Introduction

Domain semirings and Kleene algebras with domain [1, 2] yield particularly simple program verification formalisms in the style of dynamic logics, algebras of predicate transformers or boolean algebras with operators (which are all related).

There are two kinds of axiomatisation. Both are inspired by properties of the domain operation on binary relations, but target other computationally interesting models such as program traces or paths on digraphs as well.

The initial two-sorted axiomatisation [1] models the domain operation as a map $d : S \rightarrow B$ from an additively idempotent semiring $(S, +, \cdot, 0, 1)$ into a boolean subalgebra B of S bounded by 0 and 1. This seems natural as domain elements form powerset algebras in the target models mentioned. Yet the domain algebra B cannot be chosen freely: B must be the maximal boolean subalgebra of S bounded by 0 and 1 and equal to the set S_d of fixpoints of d in S .

The alternative, one-sorted axiomatisation [2] therefore models d as an endofunction on a semiring S that induces a suitable domain algebra on S_d —yet gener-

^aÉcole Polytechnique, France

E-mail: uli@lix.polytechnique.fr, ORCID: 0000-0001-9094-7625

^bNorwegian University of Science and Technology

E-mail: christian.johansen@ntnu.no, ORCID: 0000-0002-1525-0307

^cUniversity of Sheffield, UK

E-mail: g.struth@sheffield.ac.uk, ORCID: 0000-0001-9466-7815

^dUniversity of Warsaw, Poland

E-mail: ziemians@mimuw.edu.pl, ORCID: 0000-0001-7695-4028

ally only a bounded distributive lattice. An antidomain (or domain complementation) operation is needed to obtain boolean domain algebras.

In the model of binary relations over a set X , $+$ is set union and \cdot relational composition; 0 is the empty relation and 1 the identity relation. The domain of relation $R \subseteq X \times X$ is $\mathbf{d}(R) = \{(x, x) \mid \exists y. (x, y) \in R\}$ while its antidomain is $\mathbf{a}(R) = \{(x, x) \mid \forall y. (x, y) \notin R\}$. In the path model over a directed graph $\sigma, \tau : E \rightarrow V$, the carrier set consists of all finite paths $(v_1, e_1, v_2, \dots, v_{n-1}, e_{n-1}, v_n)$ in the graph in which vertices $v_i \in V$ and edges $e_i \in E$ alternate and are compatible with the source map σ and target map τ . The operations $+$ and 0 are again \cup and \emptyset , respectively; 1 is V with elements $v \in V$ seen as paths of length 1. Extending σ and τ to paths as expected, composition $\pi_1; \pi_2$ of paths π_1 and π_2 is defined if $\tau(\pi_1) = \sigma(\pi_2)$, and it then glues on this vertex. Path composition is lifted to sets of paths as $P; Q = \{\pi_1; \pi_2 \mid \pi_1 \in P, \pi_2 \in Q, \tau(\pi_1) = \sigma(\pi_2)\}$. Finally $\mathbf{d}(P) = \{\sigma(\pi) \mid \pi \in P\}$ and $\mathbf{a}(P) = \{v \mid \forall \pi. \sigma(\pi) = v \Rightarrow \pi \notin P\}$. Other models can be found in the literature.

This note revisits the two axiomatisations mentioned above to tie some loose ends together. We describe a natural algebraic setting in which they coincide, and which has so far been overlooked. It consists of additively idempotent semirings in which the sets of all elements below 1 form boolean algebras, as is the case, for instance, in boolean monoids and boolean quantales. We further take the opportunity to discuss domain axioms for arbitrary quantales.

The restriction to such boolean settings has little impact on applications: most models of interest are powerset algebras and hence (complete atomic) boolean algebras anyway. Yet the coincidence itself does make a difference: one-sorted domain semirings are easier to formalise in interactive proof assistants and apply in program verification and correctness.

2 Domain Axioms for Semirings

First we recall the two axiomatisations of domain semirings and their relevant properties. To distinguish them, we call the first class, introduced in [1], *test dioids with domain* and the second one, introduced in [2], *domain semirings*.

We assume familiarity with posets, lattices and semirings. A *dioid*, in particular, is an idempotent semiring $(S, +, \cdot, 0, 1)$, that is, $x + x = x$ holds for all $x \in S$. Its additive monoid $(S, +, 0)$ is then a semilattice ordered by $x \leq y \Leftrightarrow x + y = y$ and with least element 0 ; multiplication preserves \leq in both arguments. (We generally omit the \cdot for multiplication.)

We write $S_1 = \{x \in S \mid x \leq 1\}$ for the set of *subidentities* in S and call S *bounded* if it has a maximal element, \top .

We call a dioid S *full* if S_1 is a boolean algebra, bounded by 0 and 1 , with $+$ as sup, \cdot as inf and an operation $(_)'$ of complementation that is defined only on S_1 .

Definition 1 ([1]). *A test dioid (S, B) is a dioid S that contains a boolean subalgebra B of S_1 —the test algebra of S —with least element 0 , greatest element 1 , in which $+$ coincides with sup and that is closed under multiplication.*

Once again we write $(-)'$ for complementation on B .

Lemma 1 ([1]). *In every test dioid, multiplication of tests is their meet.*

Lemma 2 ([1]). *Let (S, B) be a test dioid. Then, for all $x \in S$ and $p \in B$,*

1. $x \leq px \Leftrightarrow p'x = 0$,
2. $x \leq px \Leftrightarrow x \leq p\top$ if S is bounded.

Definition 2 ([1]). *A test dioid with predomain is a test dioid (S, B) with a predomain operation $d : S \rightarrow B$ such that, for all $x \in S$ and $p \in B$,*

$$x \leq d(x)x \quad \text{and} \quad d(px) \leq p.$$

It is a test dioid with domain if it also satisfies, for $x, y \in S$, the locality axiom

$$d(xd(y)) \leq d(xy).$$

Weak locality $d(xy) \leq d(xd(y))$ already holds in every test dioid with predomain. Thus $d(xd(y)) = d(xy)$ in every test dioid with domain.

It is easy to check that binary relations and sets of paths satisfy the axioms of test dioids with domain, and that $B = S_1$ in both models.

Lemma 3 ([1]). *In every test dioid (S, B) , the following statements are equivalent:*

1. (S, B, d) is a test dioid with predomain,
2. the map $d : S \rightarrow B$ on (S, B) satisfies, for all $x \in S$ and $p \in B$, the least left absorption property

$$d(x) \leq p \Leftrightarrow x \leq px, \tag{11a}$$
3. in case S is bounded, $d : S \rightarrow B$ on (S, B) is, for all $x \in S$ and $p \in B$, the left adjoint in the adjunction

$$d(x) \leq p \Leftrightarrow x \leq p\top. \tag{d-adj}$$

Interestingly, test algebras of test dioids with domain cannot be chosen ad libitum: they are formed by those subidentities that are complemented relative to the multiplicative unit [1]. This has the following consequences.

Proposition 1. *The test algebra B of a test dioid with domain (S, B, d) is the largest boolean subalgebra of S_1 .*

We write $S_d = \{x \mid d(x) = x\}$ and $d(S)$ for the image of S under d .

Lemma 4 ([2]). *Let (S, B, d) be a test dioid with domain. Then $B = S_d = d(S)$.*

Next we turn to the second type of axiomatisation.

Definition 3 ([2]). A domain semiring is a semiring S with a map $\mathbf{d} : S \rightarrow S$ such that, for all $x, y \in S$ and with \leq defined as for dioids,

$$x \leq \mathbf{d}(x)x, \tag{d1}$$

$$\mathbf{d}(x\mathbf{d}(y)) = \mathbf{d}(xy), \tag{d2}$$

$$\mathbf{d}(x) \leq 1, \tag{d3}$$

$$\mathbf{d}(0) = 0, \tag{d4}$$

$$\mathbf{d}(x + y) = \mathbf{d}(x) + \mathbf{d}(y). \tag{d5}$$

Every domain semiring is a dioid: $\mathbf{d}(1) = \mathbf{d}(1)1 = 1 + \mathbf{d}(1)1 = 1 + \mathbf{d}(1) = 1$, where the second identity follows from (d1) and the last one from (d3), therefore $1 + 1 = 1 + \mathbf{d}(1) = 1$ and finally $x + x = x(1 + 1) = x$. It follows that \leq is a partial order and that axiom (d1) can be strengthened to $\mathbf{d}(x)x = x$.

Once again it is straightforward to check that binary relations and sets of paths form domain semirings.

In a domain semiring S , \mathbf{d} induces the domain algebra: $\mathbf{d} \circ \mathbf{d} = \mathbf{d}$ and therefore $S_{\mathbf{d}} = \mathbf{d}(S)$. Moreover, $(S_{\mathbf{d}}, +, \cdot, 0, 1)$ forms a subsemiring of S , which is a bounded distributive lattice with $+$ as binary sup, \cdot as binary inf, least element 0 and greatest element 1 [2], but not necessarily a boolean algebra.

Example 1 ([2]). The distributive lattice $0 < a < 1$ is a dioid with meet as multiplication, and a domain semiring with $\mathbf{d} = id$ and therefore $S_{\mathbf{d}} = S$. \square

Proposition 2 ([2]). The domain algebra of a domain semiring S contains the largest boolean subalgebra of S bounded by 0 and 1.

Axiom (d5) implies that \mathbf{d} is order preserving: $x \leq y \Rightarrow \mathbf{d}(x) \leq \mathbf{d}(y)$. In addition, $\mathbf{d}(px) = p\mathbf{d}(x)$ for all $p \in S_{\mathbf{d}}$, $\mathbf{d}(1) = 1$, and $\mathbf{d}(\top) = 1$ if S is bounded. More importantly, (lla) can now be derived for all $p \in S_{\mathbf{d}}$ (it need not hold for $p \in S_1$) [2]; it becomes an adjunction when S is bounded.

Lemma 5. In any bounded domain semiring S , (d-adj) holds for all $p \in S_{\mathbf{d}}$.

Proof. $\mathbf{d}(x) \leq p$ implies $x = \mathbf{d}(x)x \leq px \leq p\top$ and $\mathbf{d}(x) \leq \mathbf{d}(p\top) = p\mathbf{d}(\top) = p1 = p$ follows from $x \leq p\top$. \square

As mentioned in the introduction, an antidomain operation is needed to make the bounded distributive lattice $S_{\mathbf{d}}$ boolean.

Definition 4 ([2]). An antidomain semiring is a semiring S with an operation $\mathbf{a} : S \rightarrow S$ such that, for all $x, y \in S$,

$$\mathbf{a}(x)x = 0, \quad \mathbf{a}(x) + \mathbf{a}(\mathbf{a}(x)) = 1, \quad \mathbf{a}(xy) \leq \mathbf{a}(x\mathbf{a}(\mathbf{a}(y))).$$

Antidomain models boolean complementation in the domain algebra; the domain operation can be defined as $\mathbf{d} = \mathbf{a} \circ \mathbf{a}$ in any antidomain semiring S . The second and third antidomain axioms then simplify to $\mathbf{a}(x) + \mathbf{d}(x) = 1$ and $\mathbf{a}(xy) \leq \mathbf{a}(x\mathbf{d}(y))$. The domain algebra $S_{\mathbf{d}}$ of S is the maximal boolean subalgebra of S_1 , as in Proposition 1. This leads to the following result.

Lemma 6 ([2]). *Let (S, \mathbf{a}) be an antidomain semiring. Then (S, S_d, \mathbf{d}) is a test dioid with domain.*

If the domain algebra S_d of a domain semiring S happens to be a boolean algebra, it must be the maximal boolean subalgebra of S_1 by Proposition 2, so that S is again a test dioid with $B = S_d$. Antidomain is then definable.

Lemma 7. *Every domain semiring with boolean domain algebra is an antidomain semiring.*

Proof. With $\mathbf{a} = (-)' \circ \mathbf{d}$, the first antidomain axiom follows immediately from Lemma 2(1); the remaining two axioms hold trivially. \square

Example 2. In the dioid $0 < a < 1$ from Example 1, $\mathbf{d} : 0 \mapsto 0, a \mapsto 1, 1 \mapsto 1$ defines another domain semiring with $S_d = \{0, 1\} = B$. So $S_d \subset S_1$ is the maximal boolean subalgebra in S_1 . In addition, $\mathbf{a} : 0 \mapsto 1, a \mapsto 0, 1 \mapsto 0$ defines the corresponding antidomain semiring. Finally, this dioid is a test dioid by Lemma 6 and in fact a test dioid with domain in which $B = S_d \subset S_1$. \square

As powerset algebras, relation and path domain semirings have of course boolean domain algebras with complement $x' = 1 \cap \bar{x}$, where \bar{x} denotes complementation on the entire powerset algebra. Both are therefore antidomain semirings, with the operations shown in the introduction.

We finish this section with an aside on fullness:¹ While every test dioid with domain and every antidomain semiring is full whenever $S_d = S_1$ by Proposition 1 and Lemma 6, in domain semirings, $S_d = S_1$ need not imply that S_d is boolean (Example 1) and vice versa (Example 2). A domain semiring S is therefore full precisely when S_d is boolean and equal to S_1 .

3 Coincidence Result

The results of Section 2 suggest that the two types of domain semiring coincide when the underlying dioid is full. We now spell out this coincidence.

Proposition 3. *Let (S, B, \mathbf{d}) be a test dioid with domain. Then (S, \mathbf{d}) is a domain semiring with $S_d = B$ and an antidomain semiring with $\mathbf{a} = (-)' \circ \mathbf{d}$.*

Proof. The domain semiring axioms are derivable in test dioids with domain [1]; the antidomain axioms follow by Lemma 7. Moreover, B is the maximal boolean subalgebra of S_1 by Proposition 1, and thus equal to S_d by Proposition 2 (alternatively Lemma 4). \square

We know from Lemma 6 that every antidomain semiring is a test dioid with domain. Hence, by Proposition 3, antidomain semirings and test dioids with domain are interdefinable (see also [2]). For the other converse of Proposition 3 we consider full domain semirings S where $S_d = S_1$ is a boolean algebra by Proposition 2. These are test dioids, hence (lla) can be used to define domain.

¹We are grateful to a reviewer for reminding us of this fact.

Corollary 1. *Let S be a full dioid with map $d : S \rightarrow S$. Then (lla) holds for all $x \in S$ and $p \in S_1$ if and only if the predomain axioms*

$$x \leq d(x)x \quad \text{and} \quad d(px) \leq p$$

from Definition 2 hold for all $x \in S$ and $p \in S_1$.

Proof. As S is a test dioid with $B = S_1$, Lemma 3(1) applies. \square

Lemma 8. *Let S be a full dioid with map $d : S \rightarrow S$ that satisfies (lla) for all $x \in S$ and $p \in S_1$. Then (S, S_1, d) is a test dioid with predomain and $S_d = S_1$.*

Proof. S is a test dioid with predomain by Corollary 1. $S_d \subseteq S_1$ because $d(x) \leq 1$ in any test dioid with predomain [1]. $S_1 \subseteq S_d$ because $p \leq 1$ implies $p = d(p)p \leq d(p)$ and $d(p) \leq p$ because $pp = p$, using (lla). \square

Proposition 4. *Let (S, d) be a full domain semiring. Then (S, S_d, d) is a test dioid with domain.*

Proof. If (S, d) is a full domain semiring, then (lla) is derivable and locality holds. Then (S, S_d, d) is a test dioid with predomain by Lemma 8 and therefore a test dioid with domain because of locality. \square

Our coincidence result, through which the two types of domain semirings are united, then follows easily from Propositions 3 and 4.

Theorem 1. *A full test dioid is a test dioid with domain if and only if it is a domain semiring.*

On full dioids, domain can therefore be axiomatised either equationally by the domain semiring axioms or those of test dioids with domain, or alternatively by (lla) and locality. The domain algebras of relation and paths domain semirings, in particular, are full.

In any dioid, hence in particular any domain semiring, fullness can be enforced, for instance, by requiring that every $p \in S_1$ be *complemented* within S_1 , that is, there exists an element $q \in S_1$ such that $p + q = 1$ and $qp = 0$. It then follows that S_1 is a boolean algebra [2].

Alternatively, in any test dioid with domain or any antidomain semiring, $S_d = S_1$ whenever $x \leq 1 \Rightarrow d(x) = x$, for all $x \in S$. Yet Example 2 shows that this implication does not suffice to make S_d boolean in arbitrary domain semirings.

Finally, locality need not hold in full test dioids that satisfy (lla).

Example 3. Consider the full test dioid with $S = \{0, 1, a, \top\}$ in which a and 1 are incomparable with respect to \leq , $aa = 0$, multiplication is defined by $a\top = \top a = a$ and $\top\top = \top$, and d maps 0 to 0 and every other element to 1 . Then (lla) holds, but $d(ad(a)) = d(a1) = d(a) = 1 > 0 = d(0) = d(aa)$. \square

4 Examples

The restriction to full test dioids is natural for concrete powerset algebras, like the relation and path algebras mentioned. It is captured abstractly, for instance, by boolean monoids and quantales.

A *boolean monoid* [1] is a structure $(S, +, \sqcap, \cdot, \bar{}, 0, 1, \top)$ such that $(S, +, \cdot, 0, 1)$ is a semiring and $(S, +, \sqcap, \bar{}, 0, \top)$ a boolean algebra. As all sups, infs and multiplications of subidentities stay below 1, every boolean monoid is a full bounded dioid; boolean complementation on S_1 is given by $p' = 1 \sqcap \bar{p}$ for all $p \in S_1$.

Domain can now be axiomatised as an endofunction, either equationally using the domain semiring or test dioid with domain axioms, or by the adjunction (d-adj) and locality, as in Section 3. Once again, the antidomain operation \mathbf{a} is complementation on S_1 . Theorem 1 has the following instance.

Corollary 2. *A boolean monoid is a test dioid with domain if and only if it is a domain semiring.*

Quantales capture the presence of arbitrary sups and infs in powerset algebras more faithfully. Formally, a *quantale* $(Q, \leq, \cdot, 1)$ is a complete lattice (Q, \leq) and a monoid $(Q, \cdot, 1)$ such that composition preserves all sups in its first and second argument. We write \bigvee for the sup and \bigwedge for the inf operator. We also write $0 = \bigwedge Q$ for the least and $\top = \bigvee Q$ for the greatest element of Q , and \vee and \wedge for binary sups and infs.

A quantale is *boolean* if its complete lattice is a boolean algebra. Every boolean quantale is obviously a boolean monoid, and every finite boolean monoid a boolean quantale. If Q is a boolean quantale, then Q_1 forms even a complete boolean algebra. In boolean quantales, predomain, domain and antidomain operations can therefore be axiomatised like in boolean monoids, and we obtain another instance of Theorem 1, analogous to Corollary 2, simply by replacing “boolean monoid” with “boolean quantale”.

As for domain semirings, Q_d need neither be full nor boolean in an arbitrary domain quantale: the dioids in Examples 1 and 2 are defined over finite semilattices and hence complete lattices. They are therefore quantales. In this case, the identity $\mathbf{d}(x \wedge 1) = x \wedge 1$ forces $Q_d = Q_1$, because this inequality implies $\mathbf{d}(x) = x$ for all $x \leq 1$, and in fact a domain semiring with a meet operation suffices for the proof.² In antidomain quantales, this identity thus implies fullness. Whether or how the fullness could be forced equationally in arbitrary domain semirings or antidomain semirings is left open.

5 Domain Quantales

Some loose ends remain to be tied together in this note as well:

- Does the interaction of domain with arbitrary sups and infs in quantales require additional axioms?

²Again we owe this observation to a reviewer.

- Why has domain not been axiomatised explicitly using the adjunction (d-adj), at least for boolean quantales?
- And why has domain in boolean monoids or quantales not been axiomatised explicitly by $d(x) = 1 \wedge x\top$, as in relation algebra?

This section answers these questions.

First, we consider the domain semiring axioms in arbitrary quantales and argue that additional sup and inf axioms are unnecessary.

Definition 5. A domain quantale is a quantale that is also a domain semiring.

As every quantale is a bounded dioid, the adjunction (d-adj) holds for every $p \in Q_d$. In addition, domain interacts with sups and infs as follows.

Lemma 9. In every domain quantale,

1. $d(\bigvee X) = \bigvee d(X)$,
2. $d(\bigwedge X) \leq \bigwedge d(X)$,
3. $d(x)(\bigwedge Y) = \bigwedge d(x)Y$ for all $Y \neq \emptyset$.

Proof.

1. d is a left adjoint by Lemma 5 and therefore sup-preserving. Sups over X are taken in Q ; those over $d(X)$ in Q_d .
2. $(\forall x \in X. \bigwedge X \leq x) \Rightarrow (\forall x \in X. d(\bigwedge X) \leq d(x)) \Leftrightarrow d(\bigwedge X) \leq \bigwedge d(X)$.
3. Every $y \in Y \neq \emptyset$ satisfies

$$d\left(\bigwedge d(x)Y\right) \leq d(d(x)y) = d(x)d(y) \leq d(x)$$

and therefore

$$\bigwedge d(x)Y = d\left(\bigwedge d(x)Y\right)\left(\bigwedge d(x)Y\right) \leq d(x)\left(\bigwedge d(x)Y\right) \leq d(x)\left(\bigwedge Y\right).$$

The converse inequality holds because $x(\bigwedge Y) \leq \bigwedge xY$ in any quantale. \square

If $Y = \emptyset$ in part (3) of the lemma, then $d(x)(\bigwedge Y) = d(x)\top$ need not be equal to

$$\top = \bigwedge \emptyset = \bigwedge d(x)Y.$$

In the quantale of binary relations over the set $\{a, b\}$, for instance, $R = \{(a, a)\}$, satisfies $d(R) = R$ and

$$d(R)\top = \{(a, a)\} \cdot \{(a, a), (a, b), (b, a), (b, b)\} = \{(a, a), (a, b)\} \subset \top.$$

Moreover, part (1) of the lemma implies that the domain algebra Q_d is a complete distributive lattice: $d(\bigvee d(X)) = \bigvee d(X)$ holds for all $X \subseteq Q$, so that any

sup of domain elements is again a domain element. Yet the sups and infs in Q_d need not coincide with those in Q .

Second, the adjunction $d(x) \leq p \Leftrightarrow x \leq p\top$ holds for all $p \in Q_1$ in a boolean quantale Q . General properties of adjunctions then imply that, for all $x \in Q$,

$$d(x) = \bigwedge \{p \in Q_1 \mid x \leq p\top\}.$$

Lemma 8, in turn, guarantees that this identity defines predomain explicitly on boolean quantales. Yet Example 3 rules out that it defines domain: the full test dioid from this example is, in fact, a boolean quantale; it satisfies (11a) and thus (d-adj), but violates the locality axiom of domain quantales.

Finally, we give two reasons why the relation-algebraic identity

$$d(x) = 1 \wedge x\top$$

cannot replace the domain axioms in boolean monoids and quantales.

It is too weak: In the boolean quantale $\{\perp, 1, a, \top\}$ with 1 and a incomparable and multiplication defined by $\top\top = \top$ and $aa = a\top = \top a = a$, it holds that $d(a) = \perp$ (when defined by $d(x) = 1 \wedge x\top$), yet $d(a)a = \perp a = \perp < a$. Therefore $d(x)x = x$ is not derivable from $d(x) = 1 \wedge x\top$ even in boolean quantales.

It is too restrictive: although $d(x) = 1 \wedge x\top$ obviously holds in the quantale of binary relations, it fails, for instance, in the quantale formed by the sets of (finite) paths over a digraph $\sigma, \tau : E \rightarrow V$ mentioned in the introduction. Recall that the domain elements of a set P of paths are a subset of V given by the sources of the these paths. It is then obvious that $V \cap P\top = \emptyset$ unless P contains a path of length one and $d(P) = \emptyset \Leftrightarrow P = \emptyset$, so that $d(P) = V \cap P\top$ fails for any P in which all paths have length greater than 1.

This type of argument applies to all powerset quantales in which the composition of underlying objects (here: paths) is generally length-increasing and the quantalic unit and domain elements are formed by fixed-length objects.

Acknowledgments

We would like to thank the reviewers for their very insightful comments.

References

- [1] Desharnais, J., Möller, B., and Struth, G. Kleene algebra with domain. *ACM TOCL*, 7(4):798–833, 2006. DOI: 10.1145/1183278.1183285.
- [2] Desharnais, J. and Struth, G. Internal axioms for domain semirings. *Science of Computer Programming*, 76(3):181–203, 2011. DOI: 10.1016/j.scico.2010.05.007.

Received 20th November 2020

Computing Different Realizations of Linear Dynamical Systems with Embedding Eigenvalue Assignment*

Gergely Szlobodnyik^{ab} and Gábor Szederkényi^{ac}

Abstract

In this paper we investigate realizability of discrete time linear dynamical systems (LDSs) in fixed state space dimension. We examine whether there exist different $\Theta = (A, B, C, D)$ state space realizations of a given Markov parameter sequence \mathcal{Y} with fixed B , C and D state space realization matrices. Full observation is assumed in terms of the invertibility of output mapping matrix C .

We prove that the set of feasible state transition matrices associated to a Markov parameter sequence \mathcal{Y} is convex, provided that the state space realization matrices B , C and D are known and fixed. Under the same conditions we also show that the set of feasible Metzler-type state transition matrices forms a convex subset. Regarding the set of Metzler-type state transition matrices we prove the existence of a structurally unique realization having maximal number of non-zero off-diagonal entries.

Using an eigenvalue assignment procedure we propose linear programming based algorithms capable of computing different state space realizations. By using the convexity of the feasible set of Metzler-type state transition matrices and results from the theory of non-negative polynomial systems, we provide algorithms to determine structurally different realizations. Computational examples are provided to illustrate structural non-uniqueness of network-based LDSs.

Keywords: linear dynamical systems, parameter identification, structured systems, networks, convex optimization

*This work was prepared with the professional support of the Doctoral Student Scholarship Program of the Co-operative Doctoral Program of the Ministry of Innovation and Technology financed from the National Research, Development and Innovation Fund. The second author acknowledges the support of the projects EFOP-3.6.3-VEKOP-16-2017-00002 of the European Union co-financed by the European Social Fund, and K131545 of the National Research, Development and Innovation Office — NKFIH.

^aFaculty of Information Technology and Bionics, Pázmány Péter Catholic University, Práter u. 50/a, Budapest H-1083, Hungary

^bE-mail: szlobodnyik.gergely@itk.ppke.hu, ORCID: 0000-0003-2825-0645

^cE-mail: szederkenyi@itk.ppke.hu, ORCID: 0000-0003-4199-6089

1 Introduction

Many problems in computer science and engineering involve sequences of real-valued multivariate observations. It is often assumed that observed quantities are correlated with some underlying latent (state) variables that are evolving over time. Considering linear dependencies among the latent states and the observed variables leads us to linear dynamical systems. The application of linear systems is ubiquitous, ranging from dynamical systems modeling to time series analysis, including econometrics, meteorology, telecommunication, biomedical signal processing, or social network dynamics [15, 23, 35, 30, 17].

The aim of system identification is to construct parameterized models of dynamical systems by observing their input-output trajectories [22, 37]. A popular and theoretically advantageous method for estimating the parameters of linear dynamical systems is the maximum likelihood method together with expectation maximization or numerical optimization [31, 25, 16, 10]. Although the underlying mathematical representation of linear systems is simple, due to the fact that the associated optimization problem to be solved might be non-convex, estimating their parameters could be a computationally challenging task [16]. A related problem, structural identifiability examines the theoretical possibility to uniquely determine the model parameters, assuming perfect observational data [37, 24, 3]. It turns out that even in the case of linear dynamical systems, the underlying parameters may not be uniquely determined, i.e. different parameterizations of the same model structure may provide us with the same dynamical behavior.

One can observe a growing interest in both quantitative and qualitative examination of the underlying interconnected structure of dynamical systems [7, 20, 32, 4, 34]. There is a growing importance of large scale distributed engineering systems, such as power grids, distributed computing networks and intelligent transportation networks that are composed of smaller functional subunits. The interconnected structure corresponding to the state variables has attracted much attention in the context of physico-chemical systems such as chemically interacting species composing systems biological networks: gene regulatory networks, protein-protein interaction networks, metabolic networks and signal transduction pathways [36, 38]. Analyzing the locally connected structure of social networks could help us understand how viruses and information spread across the population [5, 26, 28, 29]. Subsystems, functional units are locally connected to each other according to some physical interaction topology encoded by their differential equation based description. The distributed, locally connected structure of dynamical systems poses important requirements towards efficient computational approaches, e.g. distributed controller synthesis methods over traditional centralized control algorithms [33, 9]. It can be observed in many dynamical systems that the underlying network structure is topologically non-unique i.e., different interconnection (graph) patterns can be encoded by the same dynamical equations [1, 2]. Naturally, the non-uniqueness of the network structure implies that the dynamical system is structurally non-identifiable.

Main results: In this paper we investigate realizability and structural properties of discrete time linear time invariant dynamical systems. We examine structural implications of non-unique realizability on the interaction pattern of the state variables as they are encoded in the state transition matrix. We examine the non-uniqueness of state transition matrix of LDSs. Assuming fixed input matrix B and invertible observation matrix C we prove that the feasible set of system matrices formulate a convex set. We devote particular attention to LDSs of state transition matrices that are constrained to be of Metzler property. We prove the convexity of the feasible set of state transition matrices provided that the Metzler constraint is posed. Using the eigenvalue assignment procedure we formulate a convex optimization based procedure that can be efficiently employed to find different realizations of LDSs. Assuming the Metzler property and making use of the convexity of the feasible set of system matrices we provide algorithms capable of determining structurally different dynamically equivalent state space realizations.

2 Background and problem formulation

Mathematical notations

The notations used in the paper are summarized in Table 1 below.

Table 1: Notations

\emptyset	empty set
\mathbb{R}	the set of real numbers
$\mathbb{R}^{n \times m}$	the set of $(n \times m)$ -dimensional real valued matrices
$0^{n \times m}$	$(n \times m)$ -dimensional zero matrix
$[A]_{ij}$	the entry in the i th row of the j th column of matrix A
\ominus	subtraction operator acting on a set and a matrix, $\mathcal{A} \ominus A$ is the set given by subtracting the matrix A from all the elements of \mathcal{A}

2.1 The studied system class and its properties

A discrete time linear dynamical system (LDS) in state space representation is given by a tuple $\Theta = (A, B, C, D)$ and the associated system of difference equations (DEs) is as follows:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k), & x(0) &= x_0, \\ y(k) &= Cx(k) + Du(k), \end{aligned} \tag{1}$$

where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$ and $D \in \mathbb{R}^{p \times m}$. $x(k) \in \mathbb{R}^n$ denotes the vector of state variables, $u(k) \in \mathbb{R}^m$ and $y(k) \in \mathbb{R}^p$ are the input and the associated output of the system.

Though the solution associated to a particular parametrization Θ and initial condition x_0 is unique, the parameters characterizing the underlying dynamics are

not necessarily. There may exist distinct Θ, Θ' parametrizations of the same input-output behavior meaning that the system is not structurally identifiable.

Definition 1. A system of the form of E.q. (1) is said to be structurally (globally) identifiable, if for any admissible input $u(k)$ and $k \geq 0$ we have that

$$y(k|\Theta_1) = y(k|\Theta_2) \Rightarrow \Theta_1 = \Theta_2,$$

where $y(k|\Theta)$ denotes the output of the system E.q. (1) parametrized by Θ .

If the condition of structural identifiability does not hold, the system is said to be structurally non-identifiable.

In case of structural non-identifiability, in order to quantitatively characterize the system, it is appealing to describe the feasible set of possible parameters. A quantitative characterization of the feasible set may help us finding realizations of favorable properties, such as sparsity.

Definition 2. It is said that a tuple $\Theta' = (A', B', C', D')$ is a (dynamically equivalent) realization of a LDS of the form E.q. (1) parametrized by Θ , if Θ' provides the same input-output behavior, i.e. $y(k|\Theta') = y(k|\Theta)$ for any admissible input signal $u(k)$, $k \geq 0$.

By recursively expanding E.q. (1) one can obtain the input-output equations – a common starting point of system identification – of the following form:

$$y(k) = CA^k x(0) + \sum_{i=0}^{k-1} Y_{k-i-1} u(i) + Du(k), \quad (2)$$

where the terms $Y_{k-i-1} = CA^{k-i-1}B$ and D are called the Markov parameters of the systems which are unique descriptors of the input-output behavior and are invariant to any invertible state transformations. Since Markov parameters are unique regarding the input-output behavior, we can formulate sufficient and necessary condition of dynamical equivalence with respect to the Markov parameters as follows: a tuple $\Theta' = (A', B', C', D')$ is a dynamically equivalent realization of $\mathcal{Y} = \{Y_k = CA^k B\}_{k \geq 0}$, if it satisfies $Y_k = C' A'^k B'$ for $k \geq 0$ and $D' = D$.

2.2 Problem setup

A related problem of structural non-identifiability of LDSs is the existence of distinct, $A, A' \in \mathbb{R}^{n \times n}$ state transition matrices having different patterns in their non-zero entries, i.e. structurally different state transition matrices. Assuming that E.q. (1) describes the dynamical behavior of a network-based system, the state transition matrix A can be viewed as a weighted adjacency matrix characterizing the interactions – in terms of both the interaction pattern and the magnitudes – among the components, i.e. state variables. Such a way structural non-uniqueness of a network topology can be recast as an identification problem, namely finding structurally different n -dimensional state space realizations.

In this work we concerned with the existence different realizations of LDSs and focus on the non-uniqueness and structure of the feasible state transition matrices.

Assumptions Throughout this paper we assume that a LDS is given by a state space realization $\Theta = (A, B, C, D)$ and the matrices B , C and D are fixed over all the dynamical equivalent realizations of interest. We set C to be invertible. Regarding the initial condition we assume $x(0) = 0^n$.

By fixing the matrices B , C and D we explicitly restrict our attention to dynamically equivalent realizations with different system matrices, but fixed input and output patterns. This is particularly important in the context of network-based dynamical systems where different state transition matrices incorporate distinct interaction patterns of the system components. We note that the invertibility of C covers the case of fully observable state variables.

Making use of the Markov parameter based description together with the above assumptions, the following constraint set can be employed in order to express dynamical equivalence of different realizations:

$$CA^k B = CA'^k B, \quad k \geq 0. \quad (3)$$

One difficulty with respect to the above constraint set is that generally we have a countable set of Markov parameters $\mathcal{Y} = \{Y_k\}_{k \geq 0}$ implying infinitely many constraints of the form E.q. (3). On the other hand, the terms $CA'^k B$ are non-linear and are not convex in the entries of A' – even for stable systems of nilpotent state transition matrices – which could easily make the identification problem computationally intractable.

In this paper identifiability of the above defined class of LDSs is studied. We wish to quantitatively characterize the feasible set of state transition matrices in the studied class of LDSs. We also address the problem of determining structurally different n -dimensional realizations of a LDS given by a particular initial state space realization Θ .

3 Embedding eigenvalue assignment procedure

In this section a static full-output feedback based approach is used for stabilizing a LDS and constructing a compressed set of closed-loop Markov parameters. The procedure detailed here is known as embedding eigenvalue procedure and applied in LDS identification to recover the Markov parameters [27, 18].

Let us take a LDS of E.q. (1). By taking an arbitrary $M \in \mathbb{R}^{n \times n}$ we can reformulate Eq. (1) as follows:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + My(k) - My(k) \\ y(k) &= Cx(k) + Du(k). \end{aligned} \quad (4)$$

Then for the state equation we have

$$x(k+1) = (A + MC)x(k) + (B + MD)u(k) - My(k). \quad (5)$$

Let us introduce the following matrices and new input variable

$$\bar{A} = A + MC, \quad (6)$$

$$\bar{B} = [B + MD, -M], \quad (7)$$

$$v(k) = [u(k) \ y(k)]^\top. \quad (8)$$

Then the state space model Eq. (1) can be reformulated in the following equivalent form:

$$\begin{aligned} x(k+1) &= \bar{A}x(k) + \bar{B}v(k) \\ y(k) &= Cx(k) + Du(k). \end{aligned} \quad (9)$$

Now by recursively expanding E.q. (9) the input-output behavior can be expressed as

$$y(k) = C\bar{A}^k x(0) + \sum_{i=0}^{k-1} C\bar{A}^{i-1} \bar{B}v(k-i) + Du(k). \quad (10)$$

If M can be chosen so that $\bar{A} = A + MC$ is a stability matrix, then for the Markov parameters asymptotically we have

$$\lim_{i \rightarrow \infty} C\bar{A}^i \bar{B} = 0 \quad (11)$$

In this case, E.q. (10) can be approximated as

$$y(k) \approx \sum_{i=0}^{p-1} C\bar{A}^{i-1} \bar{B}v(k-i) + Du(k) \quad (12)$$

for a suitably high $p \in \mathbb{N}$. In particular, if $A + MC$ is set to be nilpotent, then $(A + MC)^n = 0^{n \times n}$ holds. Note that such a stabilizing M matrix exists, if the system E.q. (1) is observable. Such a way the countable set of Markov parameters $\mathcal{Y} = \{CB, CAB, CA^2B, \dots\}$ is compressed to a finite set $\bar{\mathcal{Y}} = \{C\bar{B}, C\bar{A}\bar{B}, C\bar{A}^2\bar{B}, \dots, C\bar{A}^{n-1}\bar{B}\}$. For the compressed Markov parameters we introduce the notation $\bar{Y}_k = C\bar{A}^k \bar{B}$.

It can be shown that the system Markov parameters \mathcal{Y} can be uniquely recovered from that of the closed-loop system $\bar{\mathcal{Y}}$ of E.q. (4) as follows: [27, 18]:

$$Y_k = \bar{Y}_k^{(1)} + \sum_{i=0}^{k-1} \bar{Y}_i^{(2)} Y_{k-i-1} + \bar{Y}_k^{(2)} D, \quad k \geq 1, \quad (13)$$

where

$$\bar{Y}_k = C\bar{A}^k \bar{B} = \begin{bmatrix} C(A+MC)^k (B+MD) & -C(A+MC)^k M \end{bmatrix} = [\bar{Y}_k^{(1)} \ \bar{Y}_k^{(2)}] \quad (14)$$

for $k \geq 1$.

4 Representing different realizations using a compressed set of Markov parameters

In this section we show that dynamic equivalence of n -dimensional LDS realizations can be traced back to a finite set of linear equations. We make use of the eigenvalue assignment procedure, such a way instead of a countable set of Markov parameters \mathcal{Y} one can consider a compressed set of n Markov parameters $\bar{\mathcal{Y}}$ of a (stabilized) closed-loop system. By an inductive proof a linear reformulation of the non-convex equations of E.q. (3) is provided. We also show the existence of a bijection between the original state space realizations and the closed-loop system realizations.

Making use of the embedding eigenvalue assignment procedure we can obtain a finite set of compressed system descriptors $\bar{\mathcal{Y}} = \{\bar{Y}_k\}_{k=0}^{n-1}$ which is unique with respect to the closed-loop system. Finding different realizations of $\bar{\mathcal{Y}}$ can be recast in the form of a finite set of non-linear equations:

$$C\bar{A}'^k\bar{B} = C\bar{A}^k\bar{B}, \quad k = 1, \dots, n. \tag{15}$$

Note that the nilpotency of \bar{A} implies that the n th equation is equivalent to $C\bar{A}'^n\bar{B} = 0^{n \times (n+m)}$, furthermore, the invertability of C means that $C\bar{A}'^k\bar{B} = 0^{n \times (n+m)}$ for $k \geq n$.

E.q. (15) together with $C\bar{A}'^n\bar{B} = 0^{n \times (n+m)}$ provide us with a finite set of constraints to be satisfied by all the dynamically equivalent realizations $(\bar{A}', \bar{B}, C, D)$ of $\bar{\mathcal{Y}}$. However, E.q. (15) contains high nonlinearities in \bar{A}' which makes the identification problem non-convex and computationally intractable.

Proposition 1. *Let us consider a LDS of Markov sequence \mathcal{Y} with a state space representation $\Theta = (A, B, C, D)$. Assume that $\exists C^{-1}$. Then we have that*

$$CA^k B = CA'A^{k-1}B, \quad k \geq 1 \tag{16}$$

holds for any feasible n -dimensional realization $\Theta' = (A', B, C, D)$ of \mathcal{Y} .

Proof. Let us assume that $\Theta' = (A', B, C, D)$ is a dynamically equivalent realization of \mathcal{Y} we have that

$$CA^k B = CA'^k B, \quad k \geq 0.$$

For $k = 1$

$$CAB = CA'B = CA'A^0B.$$

By induction assume that for some $k > 1$ the equation $CA^k B = CA'^k B$ holds. Then

$$CA^{k+1}B = CA'^{k+1}B = CA'A'^k B = CA'C^{-1}CA'^k B = CA'C^{-1}CA^k B = CA'A^k B.$$

□

Making use of Proposition 1 the constraint set defined by E.q. (3) can be equivalently reformulated as $CA^k B = CA'A^{k-1}B$ for $k \geq 0$ which are linear in A' . Similarly one can formulate a finite set of linear constraints for the closed-loop system:

$$C\bar{A}^k\bar{B} = C\bar{A}'\bar{A}^{k-1}\bar{B}, \quad k = 1, \dots, n \quad (17)$$

By equipping E.q. (17) with a linear objective function $c : \mathbb{R}^{n \times n} \mapsto \mathbb{R}$, we obtain a linear program of the decision variables A' , e.g.:

$$\begin{cases} \max c(\bar{A}') \\ \text{subject to} \\ C\bar{A}^k\bar{B} = C\bar{A}'\bar{A}^{k-1}\bar{B}, \quad k = 1, \dots, n \end{cases} \quad (18)$$

Such a way a computational model is provided to determining dynamically equivalent realizations (\bar{A}, \bar{B}, C, D) of the closed-loop system $\bar{\mathcal{Y}} = \{C\bar{A}^k\bar{B}\}_{k=1}^n$. Furthermore, the feasible set of solutions of the linear program (18) provides all the dynamically equivalent realizations of $\bar{\mathcal{Y}}$. We note that in the optimization problem (18) the decision variables are the entries of the matrix A' , i.e. the number of decision variables is n^2 where n is the dimension of the system.

Now it can be shown that the resulted closed-loop state transition matrix \bar{A}' can be used to reconstruct an n -dimensional realization of the open loop system E.q. (1) described by the initial countable set of Markov parameters.

Proposition 2. *Let us consider a closed-loop LDS $\bar{\mathcal{Y}}$ with a state space representation $\bar{\Theta} = (\bar{A}, \bar{B}, C, D)$ so that $\bar{A}^n = 0^{n \times n}$, $\bar{A} = A + M$ and $\bar{B} = [B + MD, -M]$ for some $A, M \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^n$. Assume that there exists $\bar{A}' \in \mathbb{R}^{n \times n}$, $\bar{A}' \neq \bar{A}$ so that*

$$C\bar{A}^k\bar{B} = C\bar{A}'^k\bar{B}, \quad k = 1, \dots, n,$$

i.e. $\bar{\Theta}' = (\bar{A}', \bar{B}, C, D)$ is a dynamically equivalent realization of $\bar{\mathcal{Y}}$. Then $\Theta' = (A', B, C, D)$ is a dynamically equivalent realization of $\mathcal{Y} = \{CA^k B\}_{k \geq 0}$, where $A' = \bar{A}' - M$.

Proof. For the sake of convenience we introduce the following notations

$$\begin{aligned} Y_k(A) &= CA^k B, & \bar{Y}_k(A) &= C\bar{A}^k\bar{B}, \\ \bar{Y}_k^{(1)}(A) &= C(A + MC)^k(B + MD), & \bar{Y}_k^{(2)}(A) &= -C(A + MC)^k M \end{aligned}$$

to emphasize the dependence on a particular A . E.q. $C\bar{A}^k\bar{B} = C\bar{A}'^k\bar{B}$ implies that $\bar{Y}_k^{(1)}(A) = \bar{Y}_k^{(1)}(A')$ and $\bar{Y}_k^{(2)}(A) = \bar{Y}_k^{(2)}(A')$ hold for $k \geq 1$. Since $Y_0 = CB$ does not depend on the state transition matrix, applying recursively E.q. (13) for $k \geq 1$ we obtain that $Y_k(A) = Y_k(A')$, $k \geq 0$, i.e. $\Theta' = (A', B, C, D)$ is a dynamically equivalent realization of \mathcal{Y} . \square

5 The geometrical structure of the set of feasible system matrices

In this section we consider the set of feasible n -dimensional system matrices. We prove that for fixed B, C and D parameters, the set of feasible system matrices with respect to any \mathcal{Y} Markov sequence is convex. The set of feasible system matrices is denoted as follows:

$$\mathcal{A}(\mathcal{Y}, B, C, D) = \left\{ A \mid A \in \mathbb{R}^{n \times n}, (A, B, C, D) \text{ is a realization of } \mathcal{Y} = \{Y_k\}_{k \geq 0} \right\}. \tag{19}$$

Proposition 3. *Let us consider a countable sequence of Markov parameters \mathcal{Y} realizable by a state space realization (A, B, C, D) of order n and denote $\mathcal{A}(\mathcal{Y}, B, C, D)$ the set of feasible n -dimensional system matrices as it is defined by E.q. (19). Assume that C is invertible. Then \mathcal{A} is convex.*

Proof. Let us consider two matrices $A_1, A_2 \in \mathbb{R}^{n \times n}$ so that (A_1, B, C, D) and (A_2, B, C, D) are realizations of \mathcal{Y} . From Proposition 1 it follows that for any $a \in (0, 1)$

$$\begin{aligned} CA^k B &= aCA^k B + (1 - a)CA^k B = \\ aCA_1 A^{k-1} B + (1 - a)CA_2 A^{k-1} B &= C \left(aA_1 + (1 - a)A_2 \right) A^{k-1} B, \quad k \geq 1 \end{aligned} \tag{20}$$

In the sequel for the sake of convenience we use the notation $\hat{A} = aA_1 + (1 - a)A_2$.

Now by induction we prove that $CA^k B = C\hat{A}^k B$ for $k \geq 1$.

For $l = 1$ we have

$$CAB = C \left(aA_1 + (1 - a)A_2 \right) B.$$

Using the inductive assumption $CA^l B = C\hat{A}^l B$ for general l we obtain that

$$\begin{aligned} CA^{l+1} B &= C\hat{A}\hat{A}^l B = C\hat{A}C^{-1}CA^l B = \\ C\hat{A}C^{-1}C\hat{A}^l B &= C\hat{A}^{l+1} B \end{aligned}$$

We have that any convex combination $aA_1 + (1 - a)A_2$ results in a feasible state space realization $(aA_1 + (1 - a)A_2, B, C, D)$ of the Markov sequence \mathcal{Y} . \square

6 Characterizing structurally different system realizations

In this section we consider realizations of special structure in their state transition matrices. The off-diagonals are constrained to be non-negative. State transition matrices having non-negative off-diagonal entries are particularly important when

the purpose is to model networks of interacting components: non-zero off-diagonal entries could represent the magnitude of interactions while negative diagonals may incorporate to information or mass leakage. Positive systems – in which all the entries of the state transition matrix are constrained to be non-negative – compose a widely-studied class of linear time invariant systems with the above structural properties [8]. Discrete time linear compartmental models – having many applications in modeling biological systems – also satisfy the above non-negativity condition [14, 13]. Social networks provide an important application field of modeling discrete time dynamical systems defined on networks [26, 28, 29, 21]. The DeGroot and Friedkin-Johnsen models are well-known discrete time linear models of opinion dynamics and information spreading in networks where the off-diagonal entries of state transition matrices are also constrained to be non-negative [6, 11].

Formally, for a Markov sequence \mathcal{Y} we restrict our attention to realizations $\Theta = (A, B, C, D)$ so that A is Metzler, i.e. $[A]_{ij} \geq 0$ for $i \neq j$. Then the feasible set of state transition matrices can be defined as follows:

$$\mathcal{A}^p(\mathcal{Y}, B, C, D) = \left\{ A \mid \begin{array}{l} [A]_{ij} \geq 0 \text{ for } i, j = 1, \dots, n, i \neq j, \\ (A, B, C, D) \text{ is a realization of } \mathcal{Y} \end{array} \right\} \quad (21)$$

Note that the convexity of $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ is guaranteed as a corollary of Proposition 3 which can be seen as follows. For any $A_1, A_2 \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$, the convex combination $aA_1 + (1-a)A_2$ with $a \in (0, 1)$ is a feasible state transition matrix in $\mathcal{A}(\mathcal{Y}, B, C, D)$. Since a convex combination is a linear combination with non-negative coefficients, the sign of the off-diagonal entries remain non-negative, i.e. $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ is convex.

Now with respect to the set $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ we identify matrices having distinguished structural properties and show how they relate to all the other feasible state transition matrices.

In order to ease the discussion of structural properties state transition matrices, we introduce a simple graph-based description of LDSs with state transition matrices of Metzler-type using the analogy of influence graphs in the literature of positive systems [8]. Considering a state transition matrix $A \in \mathbb{R}^{n \times n}$, the associated directed graph representation $G(A) = (E, V)$ is defined as follows. V , the set of nodes corresponds to the set of states of the associated LDS. E , the set of edges represents the influences between state variables, i.e. $(i, j) \in E$ if and only if $[A]_{ij} > 0$. Such a way $G(A)$ provides a unique description of the structure of A .

In the sequel the term structure of a state transition matrix $A \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ refers to the structure (topology) of the associated directed graph representation $G(A)$ as it is defined above.

Definition 3. *Let us consider a LDS \mathcal{Y} with fixed $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{n \times n}$ and $D \in \mathbb{R}^{n \times m}$. A matrix $A \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ is called dense (sparse) state transition matrix if it contains maximal (minimal) number of non-zero off-diagonal entries.*

Then the associated realization $\Theta = (A, B, C, D)$ is said to be a dense (sparse) realization.

Definition 4. Let us consider a LDS \mathcal{Y} with fixed $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{n \times n}$ and $D \in \mathbb{R}^{n \times m}$. A state transition matrix $A \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ is said to have superstructure property, if its graph representation $G(A)$ contains the graph representations of all other feasible Metzler system matrices as subgraphs, formally $G(A') \subseteq G(A) \forall A' \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$.

It can be shown that a dense realization provides a superstructure with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$.

Proposition 4. Let us consider a LDS of Markov parameters \mathcal{Y} with fixed $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{n \times n}$ and $D \in \mathbb{R}^{n \times m}$ state space realization matrices. Any dense state transition matrix $A^d \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ is of superstructure property.

Proof. Assume that there exists a dense state transition matrix $A^d \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ so that A^d has no superstructure property. Then it follows that there exists a state transition matrix $A \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ for which there is an index-pair (i, j) , $i \neq j$ so that $[A]_{ij} > 0$, but $[A^d]_{ij} = 0$. The convexity of $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ guarantees that for any $a \in (0, 1)$ the resulted matrix $A' = aA + (1-a)A^d$ provides a dynamically equivalent realization with non-negative off-diagonal entries, i.e. $A' \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$. Such a way we obtained a state transition matrix A' having more non-zero off-diagonal entries, than A^d has, which is contradiction. \square

Corollary 1. Let us consider a Markov sequence \mathcal{Y} . For any $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$ and $D \in \mathbb{R}^{p \times m}$, there exists a structurally unique state transition matrix A^d having maximal number of non-zero off-diagonal entries with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$.

7 Computational framework for finding structurally different realizations

In this section first we assume a state space realization $\Theta = (A, B, C, D)$ so that its respective Markov parameter sequence \mathcal{Y} is of finite-length, i.e. $CA^k B = 0^{n \times m}$, $k \geq p$ for some finite p . Examining the realizability of finite-length Markov sequences can be motivated by a partial realization problem or realizability analysis of stable and damping systems having only a finite number of non-zero Markov parameters [27, 12].

Algorithms for determining structurally different realizations of LDS with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ are provided. Making use of the convexity of $\mathcal{A}^p(\mathcal{Y}, B, C, D)$, we adopt algorithms proposed for mass action law kinetic systems and show that structurally different realizations regarding the feasible set of Metzler system matrices $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ can be efficiently obtained [1]. We prove that a dense state transition matrix A^d in $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ can be computed in polynomial time using a

convex optimization based procedure. Then it can be also shown that all the structurally different realizations of $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ can be determined by iteratively computing constrained dense realizations.

Finally we show that using the eigenvalue assignment procedure, the proposed algorithms can be extended to compute structurally different realizations of LDSs of arbitrary Markov parameter sequences.

7.1 Algorithm for computing dense realization

Here we provide an algorithm capable of finding a dense realization with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ in polynomial time, given that \mathcal{Y} is a finite sequence. The correctness of the algorithm follows from the convexity of $\mathcal{A}^p(\mathcal{Y}, B, C, D)$. First we define a subroutine denoted by **FindRealization** in order to determine feasible state transition matrices.

FindRealization: $(\Theta = (A, B, C, D), L, H)$: returns a tuple (A', P) so that $A' \in \mathbb{R}^{n \times n}$ is a feasible state transition matrix of Metzler-type, i.e. $A' \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$, and the objective function $\sum_{(i,j) \in H} [A']_{ij}$ is maximized by A' , where H is a set of index pairs. L denotes a set of index pairs so that $[A']_{ij} = 0$, if $(i, j) \in L$. Formally, A' is obtained as the solution of the following linear program:

$$\left\{ \begin{array}{l} \max \sum_{(i,j) \in H} [A']_{ij} \\ \text{subject to} \\ CA^k B = CA' A^{k-1} B, \quad k = 1, \dots, p \\ [A']_{ij} = 0, \quad (i, j) \in L \end{array} \right. \quad (22)$$

P denotes the set of ordered pairs encoding the non-zero pattern of A' so that $(i, j) \in P$ iff $[A']_{ij} > 0$. If feasible realization A' does not exist, it returns $(\mathbf{0}^{n \times n}, \emptyset)$.

Next we introduce Algorithm 1 (**FindDenseRealization**) for finding a dense dynamically equivalent realization, given a state space model $\Theta = (A, B, C, D)$.

Proposition 5. *The state transition matrix A^d returned by algorithm **FindDenseRealization** $(\Theta = (A, B, C, D), L)$ provides a dynamically equivalent realization $\Theta' = (A', B, C, D)$ of the LDS with Markov parameters $\mathcal{Y} = CA^k B$, $k = 1, \dots, p$. Furthermore, A^d is dense among all the state transition matrices in $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ satisfying the zero-constraints defined by L . A^d is computed in polynomial time.*

7.2 Algorithm for computing all structurally different realizations

Here we describe an algorithm capable of determining all structurally different realizations of any LDS $\Theta = (A, B, C, D)$ with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$, given that $\mathcal{Y} = \{CA^k B\}_{k=0}^p$ with $p > 0$ finite. Making use of Algorithm **FindDenseRealization** described in the previous section, the proposed computational method

Algorithm 1 FindDenseRealizationInput: $\Theta = (A, B, C, D), L$ Output: *Result*

```

1:  $H \leftarrow \{1, \dots, n^2 - n\}$ 
2:  $P \leftarrow H$ 
3:  $A^d \leftarrow \mathbf{0}^{n \times n}$ 
4:  $loops \leftarrow 0$ 
5: while TRUE do
6:    $(A', P) \leftarrow \text{FindRealization}(\Theta = (A, B, C, D), L, H)$ 
7:   if  $P \neq \emptyset$  then
8:     BREAK
9:   end if
10:   $A^d \leftarrow A^d + A'$ 
11:   $H \leftarrow H \setminus P$ 
12:   $loops \leftarrow loops + 1$ 
13: end while
14: if  $A_d \neq \mathbf{0}^{n \times n}$  then
15:    $A^d \leftarrow \frac{A^d}{loops}$ 
16:   return  $A^d$  // Result is a dense realization.
17: else
18:   return -1 // There is no feasible realization.
19: end if

```

iteratively finds constrained dense realizations. Such a way all distinct structure can be obtained.

Assuming a fixed ordering of the state variables, we introduce the notation \mathcal{R} to denote the set of binary sequences of length $(n \times n) - n$ encoding the structure of non-zero off-diagonal patterns of the system matrices. The i 'th entry of $R \in \mathcal{R}$ is denoted by $R[i]$. An edge e is in the graph $G(A)$ iff there exists an index $i \in \{1 \dots |E(G(A))|\}$ for which $e = e_i$ and $R[i] = 1$.

We introduce the array *Exist* of $2^{|\mathcal{R}|}$ binary variables such that $Exist[R] = 1$ iff there exists a dynamically equivalent realization encoded by the sequence $R \in \mathcal{R}$.

A stack S is employed to temporarily store tuples of the form (R, k) with $R \in \mathcal{R}$ and $k \in \mathbb{N}$. The command 'push (R, k) into S ' pushes the tuple (R, k) into S , while 'pop from S ' returns the last tuple (R, k) .

We say that the binary relation $=_k$ holds between the sequences $R, W \in \mathcal{R}$ ($R =_k W$) if for $i = 1 \dots k$, $R[i] = W[i]$. The equivalence class of the relation $=_k$ for which R is a representative element is denoted by $C_k(R)$. Note that for an equivalence class more representative elements may exist.

The following subroutines are employed in the algorithm:

1. **FindDenseRealizationSequence**($\Theta = (A, B, C, D), R, k, i$): computes a dense state transition matrix A^d with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$, given a se-

quence $R \in \mathcal{R}$ and $k, i \in \mathbb{N}$. It returns a feasible state transition matrix $A \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ and the associated binary sequence $W \in \mathcal{R}$ so that $W =_k R$ and for every $W[j] = 0$ for $j = k + 1, \dots, i$. If such a reaction does not exist returns -1.

Note that **FindDenseRealizationSequence** can be implemented by means of **FindDenseRealization**.

2. **FindNextOne**(R, k) returns the smallest index i for which $k < i$ and $R[i] = 1$. If $R[i] = 0$ for all $k < i$ then it returns $z + 1$.

Algorithm 2 FindAllRealizations

 Inputs: $\Theta = (A, B, C, D)$

 Output: *Exist*

```

1:  $D \leftarrow \text{FindDenseRealization}(\Theta = (A, B, C, D), \emptyset)$ 
2: push  $(D, 0)$  into  $S$ 
3:  $\text{Exist}[D] \leftarrow 1$ 
4: while  $\text{size}(S) > 0$  do
5:    $(R, k) \leftarrow \text{pop}$  from  $S$ 
6:    $i \leftarrow \text{FindNextOne}((R, k))$ 
7:   if  $i < z$  then
8:     push  $(R, i)$  into  $S$ 
9:   end if
10:  while  $i < z$  do
11:     $(A', W) \leftarrow \text{FindDenseRealizationSequence}(\Theta = (A, B, C, D), R, k, i)$ 
12:    if  $W < 0$  then
13:      BREAK
14:    else
15:       $i \leftarrow \text{FindNextOne}(W, i)$ 
16:       $\text{Exist}[W] \leftarrow 1$ 
17:      if  $i < z$  then
18:        push  $(W, i)$  into  $S$ 
19:      end if
20:    end if
21:  end while
22: end while

```

Proposition 6. Algorithm **FindAllRealizations**($\Theta = (A, B, C, D)$) determines all structurally different dynamical equivalent state transition matrices of a LDS given by $\Theta = (A, B, C, D)$ with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$, provided that $\mathcal{Y} = \{CA^k B\}_{k=0}^p$ for some finite $p > 0$.

7.3 Extension to arbitrary LDS

This section extends the aforementioned results in order to find structurally different realizations of arbitrary LDS. We consider a LDS $\Theta = (A, B, C, D)$ so that there are no constraints on $\mathcal{Y} = \{CA^k B\}_{k \geq 0}$. Assuming that the pair (A, C) is observable, the eigenvalue assignment procedure can be employed. Then there exists $M \in \mathbb{R}^{n \times n}$ so that $\bar{A} = A + M$ is nilpotent, i.e. $\bar{A}^n = 0$.

Consider the linear program

$$\begin{cases} \max \sum_{(i,j) \in H} [\bar{A}']_{ij} \\ \text{subject to} \\ C\bar{A}^k B = C\bar{A}'A^{k-1}B, \quad k = 1, \dots, n \\ [\bar{A}']_{ij} \geq [M]_{ij}, \quad i, j = 1, \dots, n, \quad i \neq j \\ [\bar{A}']_{ii} \leq [M]_{ii}, \quad i = 1, \dots, n \\ [\bar{A}']_{ij} = 0, \quad (i, j) \in L \end{cases} \quad (23)$$

Given a solution \bar{A}' of the linear program E.q. (23), Proposition 2 guarantees that $A' = \bar{A}' - M$ provides a dynamically equivalent realization of the system $\Theta = (A, B, C, D)$ and $A' \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$. Now we replace the linear program of E.q. (22) with E.q. (23) in **FindRealization** so that it returns (A', P) where $A' + M = \bar{A}'$ is the solution of E.q. (23) and P is as it is defined above. Then we have that the resulted algorithms **FindDenseReal** and **FindAllRealizations** determine a set of matrices \mathcal{A} for which $\mathcal{A} \ominus M$ defines a set of structurally different realizations of $\Theta = (A, B, C, D)$. For each $\bar{A}' \in \mathcal{A}$, we have that $(A' - M) \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$. This way structurally different realizations with Metzler-type state transition matrices of a LDS – of arbitrary Markov sequence – can be computed.

8 Computational examples

In this section we provide examples to illustrate structural non-uniqueness of the non-zero off-diagonal patterns of state transition matrices associated to a Markov sequence \mathcal{Y} . By some simple linear dynamical system models we show that the set of feasible state transition matrices $\mathcal{A}(\mathcal{Y}, B, C, D)$ is not necessary unique and structurally different dynamically equivalent realizations can be computed. Throughout the section we restrict our attention to realizations with system matrices of Metzler-type.

In each example, first the system is stabilized by a full-state feedback M using the algorithm of [19] in order to obtain a closed-loop system of the form of E.q. (4) with a finite sequence of non-zero Markov parameters $\bar{\mathcal{Y}}$. In Example 8.1 Algorithm 1 and 2 are employed to determine all the structurally different realizations with respect to $\mathcal{A}^p(\bar{\mathcal{Y}}, \bar{B}, C, D)$. Then Proposition 2 guarantees that structurally different realizations of the open-loop system \mathcal{Y} can be recovered by subtracting M from the closed-loop system matrices. Example 8.2 illustrate the structural non-uniqueness

of a social network equipped with a linear dynamical behavior. Indirect sparsity and density constraints are employed in order to find different realizations.

8.1 Example 1

Let us consider the following system

$$A = \begin{bmatrix} -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (24)$$

$$B = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0], \quad (25)$$

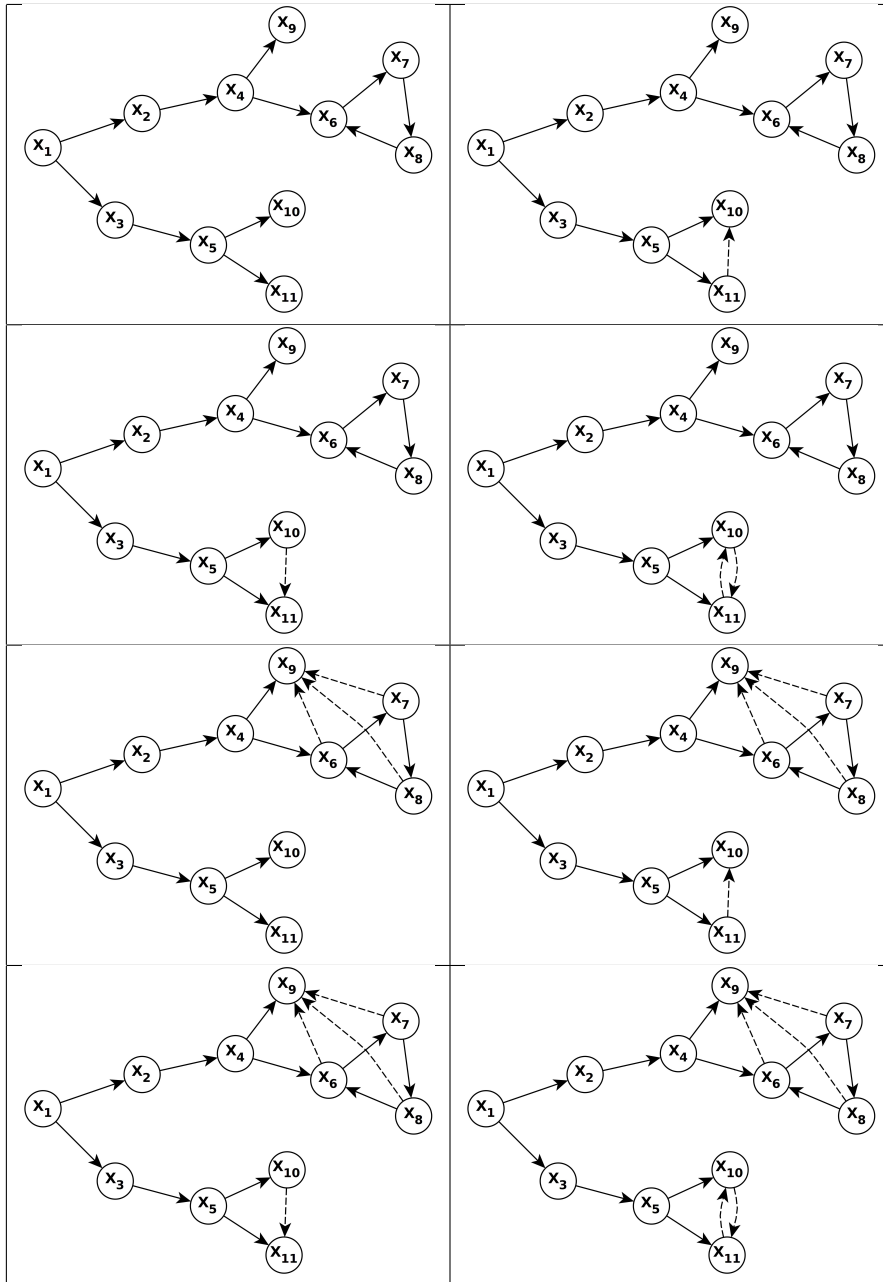
C is an $(n \times n)$ -dimensional identity matrix and $D = 0^n$.

A full-output feedback M is obtained by the algorithm [19]. Then using Algorithm 2 we determined all the structurally different closed-loop system matrices in $\mathcal{A}^p(\overline{\mathcal{Y}}, \overline{B}, C, D)$. Finally a set of structurally different state transition matrices with respect to $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ is computed by $\mathcal{A}^p(\overline{\mathcal{Y}}, \overline{B}, C, D) \ominus M$.

Figure 1 depicts the number of structurally different realizations as the function of the number of non-zero off-diagonal entries in the state transition matrix. Table 2 provides a set of structurally different realizations in $\mathcal{A}^p(\overline{\mathcal{Y}}, \overline{B}, C, D)$ as they are determined in the above described way.

$$A^d = \begin{bmatrix} -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 100 & 100 & 100 & -100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -100 & 100 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 100 & -100 \end{bmatrix} \quad (26)$$

Table 2: Graph representations of all the structurally different state transition matrices computed by **FindAllRealizations**. Non-zero entries which are not contained in the initial realization are denoted by dashed lines.



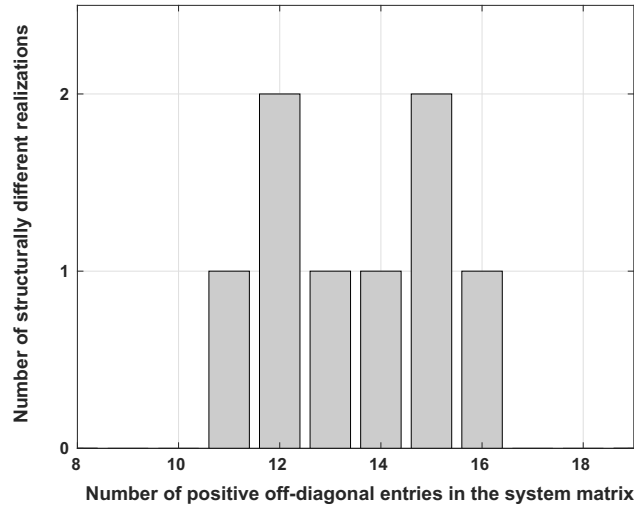


Figure 1: Structural non-uniqueness of feasible system matrices A associated to dynamically equivalent state space realizations of E.q.

The non-uniqueness of the network structure has important theoretical and practical consequences. It turned out that the system is structurally non-identifiable, that is the same dynamical behavior can be realized with different parameterizations. At the same time, a computational procedure is provided to test structural non-identifiability with theoretical guarantee. Non-uniqueness of the underlying network topology is a specific case of the lack of structural identifiability: the underlying dynamical behavior is realizable with different sets of interconnections of the state variables. The existence of different network structures is particularly important if the state variables have some biological, physical or chemical meaning: the same dynamical behavior (functionality) can be implemented using different relationships between the system variables.

8.2 Example 2

The Zachary karate club network is a widely studied social network representing the interactions of 34 members outside a Karate club [39]. Here we study the information flow across the network equipped with a particular weighted directed edge set as it is depicted in Figure 2. The weighted directed edges can be uniquely encoded in the form of an adjacency matrix $A \in \mathbb{R}^{34 \times 34}$, assuming a fixed ordering of the nodes, i.e. state variables. For the entries of A see Appendix 9. With the chosen edge directions we wish to simulate the information flow from the direction of the first node, i.e. x_1 (source) to the last nodes, x_{33} and x_{34} (sinks).

We make use of the adjacency matrix A of the network to define the dynamics of information flow over the nodes and formulate a simple LDS of the form E.q. (1). The adjacency matrix A defines the state transition matrix, $[A]_{ij} > 0$ iff there

is direct information flow from node j to node i . $B \in \mathbb{R}^{34}$ is set to be zero for all the entries except for the first one which is equal to 1, i.e. $[B]_1 = 1$ and $[B]_i = 0$ for $i = 2, \dots, 34$. This way we can examine how an input signal $u(k) \in \mathbb{R}$, $k \geq 0$ – perturbing the state of the first node – propagates along all the other nodes. $C \in \mathbb{R}^{34 \times 34}$ is the identity matrix, i.e. we assume that all the state variables are observable. $D = 0^{34}$. The state variable vector $x(t) \in \mathbb{R}^{34}$, $t \geq 0$ encodes the information content of the state variables. We assume that $x(0) = 0^{34}$.

Starting with the above defined state space model $\Theta = (A, B, C)$, first we performed the eigenvalue assignment procedure. A matrix $M \in \mathbb{R}^{34 \times 34}$ is determined so that the resulting $\bar{A} = A + M$ be nilpotent. This way a stabilized closed-loop system $\bar{\Theta} = (\bar{A}, B, C)$ – having at most 34 non-zero Markov parameters – is obtained, where $\bar{B} = [B, -M]$. In order to find a dynamically equivalent realization of the stabilized system $\bar{\Theta}$ with Metzler-type state transition matrix and sparsity constraint, we solved the following optimization procedure

$$\begin{cases} \max \sum_{\substack{i,j=1 \\ i \neq j}}^{34} |[\bar{A}']_{ij}| \\ \text{subject to} \\ C\bar{A}^k B = C\bar{A}'\bar{A}^{k-1} B, \quad k = 1, \dots, 34 \\ [\bar{A}']_{ij} \geq [M]_{ij}, \quad i, j = 1, \dots, 34, \quad i \neq j \\ [\bar{A}']_{ii} \leq [M]_{ii}, \quad i = 1, \dots, 34 \end{cases} \quad (27)$$

where the entries of \bar{A}' correspond to the decision variables. Denoting the solution of (27) by \bar{A}^s , Proposition 2 guarantees that $\hat{A}^s = \bar{A}^s - M$ provides a dynamically equivalent realization of the initial system. The obtained realization (\hat{A}^s, B, C) has 78 non-zero off-diagonal entries and its graph representation $G(\hat{A}^s)$ is isomorph to that of the initial state transition matrix $G(A)$. Next a dense realization (\bar{A}^d, \bar{B}, C) is computed with respect to the closed-loop system $\bar{\Theta}$ using Algorithm 1. Proposition 2 guarantees that $\hat{A}^d = \bar{A}^d - M$ determines a dynamically equivalent realization with respect to the initial system Θ . We found that the obtained state transition matrix \hat{A}^d contains 451 non-zero off-diagonal entries. The obtained matrices \hat{A}^s and \hat{A}^d are illustrated in Figure 3.

Since $[\bar{A}^s]_{ij} \geq [M]_{ij}$ and $[\bar{A}^d]_{ij} \geq [M]_{ij}$ hold for $i, j = 1, \dots, 34, i \neq j$, the state transition matrices \hat{A}^s and \hat{A}^d are of Metzler-type. Furthermore, $[\bar{A}^s]_{ij} = [M]_{ij}$ and $[\bar{A}^d]_{ij} = [M]_{ij}$ for $i \neq j$ imply that $[\hat{A}^s]_{ij} = 0$ and $[\hat{A}^d]_{ij} = 0$, respectively, i.e. $G(\bar{A}^s)$ and $G(\bar{A}^d)$ are isomorph to $G(\hat{A}^s)$ and $G(\hat{A}^d)$, respectively. Such a way we can put indirectly sparsity and density constraints to state transition matrices of LDS having arbitrary Markov parameters. However, it is important to note that the resulted state transition matrices \hat{A}^s and \hat{A}^d are not proved to be sparse and dense with respect to the initial system Θ , i.e. there may exist dynamically equivalent realizations having less or more non-zero off-diagonal entries, respectively.

The existence of structurally different realizations implies that the Karate club network equipped with the above dynamical system model is structurally non-identifiable. The particular importance of the example is that the same dynamical

behavior, that is information propagation among the nodes, is feasible with structurally different network topologies. Different interconnections can provide the same emergent dynamical behavior described by a DTLTI system. The result underlines that a certain network topology, even in the case of linear systems, might not be a complete descriptor of the modeled process.

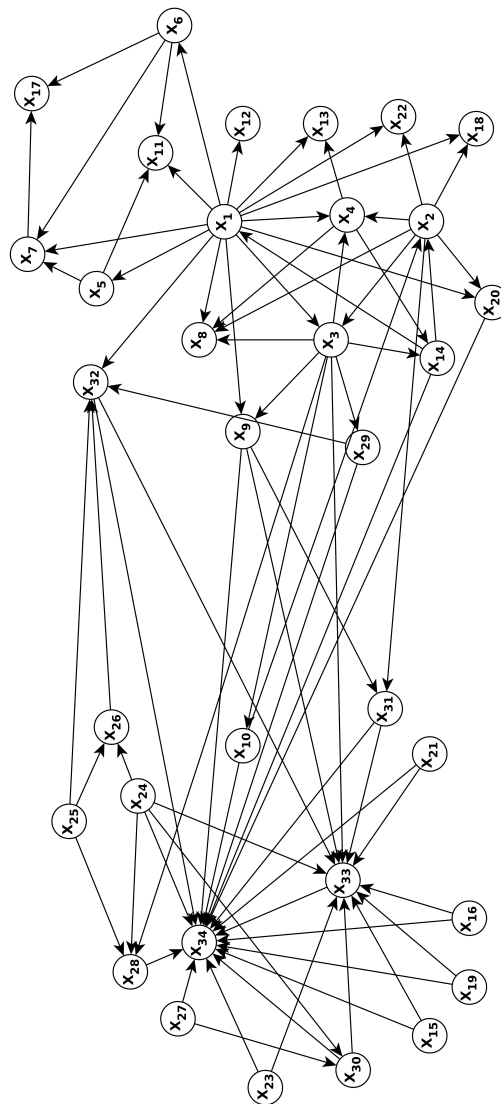


Figure 2: Illustration of Zachary's karate club network with a particular directed edge set.

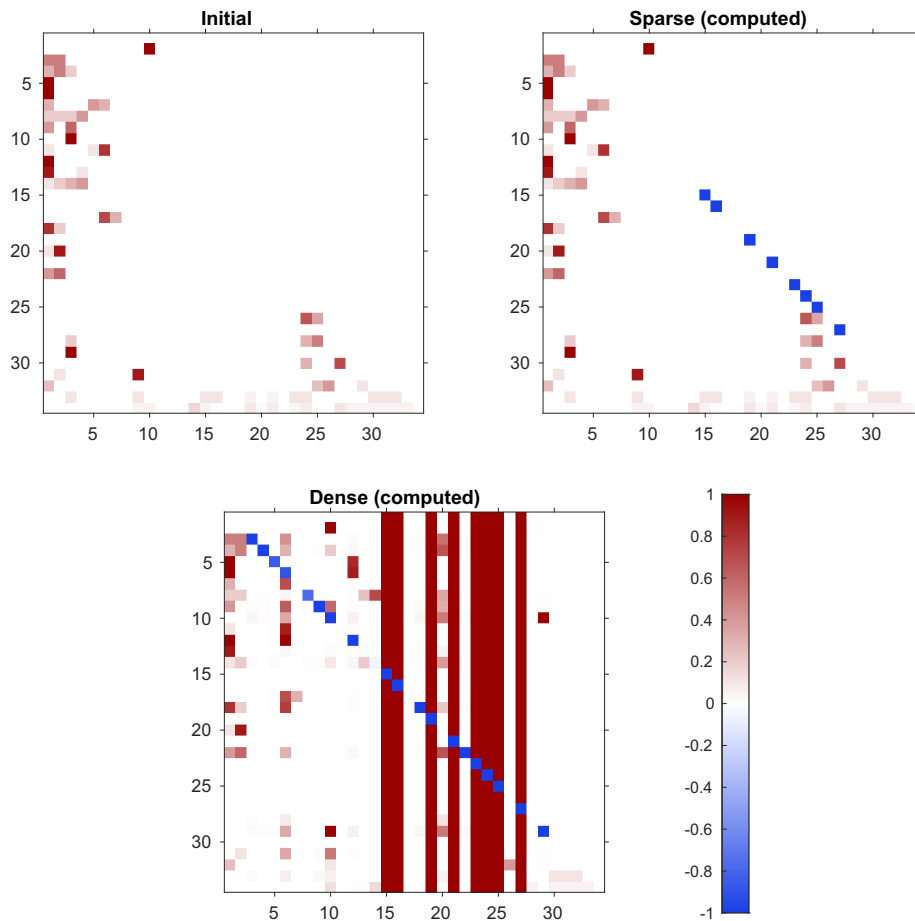


Figure 3: Graphical representation of state transition matrices associated to different realizations. **Initial:** the initial state transition matrix A . **Sparse (computed):** the state transition matrix \hat{A}^s computed by posing l_1 sparsity constraints on the off-diagonal entries (i.e. decision variables). **Dense (computed):** the state transition matrix \hat{A}^d obtained by Algorithm 1. Note that the initial and sparse matrices are equivalent in terms of the pattern of their non-zero off-diagonal entries, i.e. $G(A)$ and $G(\hat{A}^s)$ are isomorph graphs. We emphasize that sparsity, as a structural property, is understood with respect to the off-diagonal entries. The existence of structurally different state transition matrices implies that the same information propagation dynamics can emerge in structurally different networks.

9 Conclusion

In this paper we considered realizability of discrete time linear dynamical systems. Throughout the paper it is assumed that a LDS is given by a Markov parameter sequence \mathcal{Y} and that the state space realization matrices B , C and D are known and fixed. Under these assumptions the existence of different realizations of \mathcal{Y} is equivalent to the existence of distinct state transition matrices of the same dimension that provides the same sequence \mathcal{Y} . Assuming that the state space realization matrix C is invertible, we quantitatively characterized the set of feasible state space realizations. It is proved that the set of state transition matrices $\mathcal{A}(\mathcal{Y}, B, C, D)$ associated to a Markov sequence \mathcal{Y} is convex, given B , C and D matrices. Under the same conditions it is also shown that the subset of Metzler-type system matrices $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ is convex. Furthermore, we proved that there exists a structurally unique state transition matrix $A^d \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ of maximal number of off-diagonal entries whose respective graph representation $G(A^d)$ contains that of any other feasible state transition matrix in $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ as subgraph.

Making use of the eigenvalue assignment procedure, we reformulated dynamical equivalence of state space realizations in terms of a finite set of linear constraints in the entries of the state transition matrix. This way we proposed a convex optimization based algorithm that can be used to find different realizations of any Markov sequence. Since the existence of different system matrices implies structural non-identifiability of the underlying dynamical system, this way non-identifiability of LDSs can be validated in fixed state space dimension in polynomial time. By making use of the convexity of $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ and adopting results from the field of non-negative polynomial systems, we provided algorithms that can determine structurally different realizations of LDSs with respect to Metzler-type state transition matrices. Representative examples are presented in order to illustrate that dynamically equivalent realizations of LDSs are not necessary structurally unique, i.e. there may exist structurally different realizations of the same LDS even in the case of fixed B , C and D state space realization matrices.

Acknowledgement

The authors thank the anonymous Reviewers for their constructive comments.

References

- [1] Ács, Bernadett, Szederkényi, Gábor, Tuza, Zsolt, and Tuza, Zoltán A. Computing all possible graph structures describing linearly conjugate realizations of kinetic systems. *Computer Physics Communications*, 204:11–20, 2016. DOI: 10.1016/j.cpc.2016.02.020.
- [2] Ács, Bernadett, Szlobodnyik, Gergely, and Szederkényi, Gábor. A computational approach to the structural analysis of uncertain kinetic systems. *Com-*

- puter Physics Communications*, 228:83–95, 2018. DOI: 10.1016/j.cpc.2018.03.002.
- [3] Bellman, Ror and Åström, Karl Johan. On structural identifiability. *Mathematical Biosciences*, 7(3-4):329–339, 1970. DOI: 10.1016/0025-5564(70)90132-x.
- [4] Bento, José, Ibrahimi, Morteza, and Montanari, Andrea. Learning networks of stochastic differential equations. In *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, pages 172–180, 2010.
- [5] Boccaletti, Stefano, Latora, Vito, Moreno, Yamir, Chavez, Martin, and Hwang, D-U. Complex networks: Structure and dynamics. *Physics Reports*, 424(4-5):175–308, 2006. DOI: 10.1016/j.physrep.2005.10.009.
- [6] DeGroot, Morris H. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974. DOI: 10.1080/01621459.1974.10480137.
- [7] Dion, Jean-Michel, Commault, Christian, and Van der Woude, Jacob. Generic properties and control of linear structured systems: a survey. *Automatica*, 39(7):1125–1144, 2003. DOI: 10.1016/s0005-1098(03)00104-3.
- [8] Farina, Lorenzo and Rinaldi, Sergio. *Positive Linear Systems: Theory and Applications*, volume 50. John Wiley & Sons, 2000. DOI: 10.1002/9781118033029.
- [9] Fattahi, Salar, Fazelnia, Ghazal, Lavaei, Javad, and Arcak, Murat. Transformation of optimal centralized controllers into near-globally optimal static distributed controllers. *IEEE Transactions on Automatic Control*, 64(1):66–80, 2018. DOI: 10.1109/tac.2018.2829473.
- [10] Fattahi, Salar, Matni, Nikolai, and Sojoudi, Somayeh. Learning sparse dynamical systems from a single sample trajectory. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2682–2689. IEEE, 2019. DOI: 10.1109/cdc40024.2019.9029815.
- [11] Friedkin, Noah E. and Johnson, E. C. Influence networks and opinion change. *Advances in Group Processes*, 16(1):1–29, 1999.
- [12] Gragg, William B. and Lindquist, Anders. On the partial realization problem. *Linear Algebra and its Applications*, 50:277–319, 1983. DOI: 10.1016/0024-3795(83)90059-9.
- [13] Haddad, Wassim M., Chellaboina, Vijay Sekhar, and August, Elias. Stability and dissipativity theory for discrete-time non-negative and compartmental dynamical systems. *International Journal of Control*, 76(18):1845–1861, 2003. DOI: 10.1080/00207170310001635400.

- [14] Haddad, Wassim M., Chellaboina, VijaySekhar, and Hui, Qing. *Nonnegative and compartmental dynamical systems*. Princeton University Press, 2010. DOI: 10.1515/9781400832248.
- [15] Hamilton, James. *Time Series Analysis*. Princeton: Princeton University Press, 1994.
- [16] Hardt, Moritz, Ma, Tengyu, and Recht, Benjamin. Gradient descent learns linear dynamical systems. *Journal of Machine Learning Research*, 29:1–44, 2018.
- [17] Huang, Zhaoxia, Liu, Jun, and Qian, Fucui. Interval state estimation of linear multicellular systems. *Complexity*, 2019, 2019. DOI: 10.1155/2019/6438054.
- [18] Juang, Jer-Nan, Phan, Minh, Horta, Lucas G, and Longman, Richard W. Identification of observer / Kalman filter Markov parameters - theory and experiments. *Journal of Guidance, Control, and Dynamics*, 16(2):320–329, 1993. DOI: 10.2514/3.21006.
- [19] Kautsky, Jaroslav, Nichols, Nancy K, and Van Dooren, Paul. Robust pole assignment in linear state feedback. *International Journal of Control*, 41(5):1129–1155, 1985. DOI: 10.1080/0020718508961188.
- [20] Lin, Ching-Tai. Structural controllability. *IEEE Transactions on Automatic Control*, 19(3):201–208, 1974. DOI: 10.1109/tac.1974.1100557.
- [21] Liu, Ji, Ye, Mengbin, Anderson, Brian D. O., Basar, Tamer, and Nedic, Angelia. Discrete-time polar opinion dynamics with heterogeneous individuals. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 1694–1699. IEEE, 2018. DOI: 10.1109/cdc.2018.8619071.
- [22] Ljung, Lennart. *System Identification: Theory for the User*. Prentice-Hall, Inc., USA, 1986.
- [23] Ljung, Lennart and Glad, Torkel. *Modeling of dynamic systems*. Prentice-Hall, 1994.
- [24] Ljung, Lennart and Glad, Torkel. On global identifiability for arbitrary model parametrizations. *Automatica*, 30(2):265–276, 1994. DOI: 10.1016/0005-1098(94)90029-9.
- [25] Martens, James. Learning the linear dynamical system with ASOS. In *Proceedings of the 27th International Conference on Machine Learning (ICML'10)*, pages 743–750. Citeseer, 2010.
- [26] Parsegov, Sergey E., Proskurnikov, Anton V., Tempo, Roberto, and Friedkin, Noah E. Novel multidimensional models of opinion dynamics in social networks. *IEEE Transactions on Automatic Control*, 62(5):2270–2285, 2016. DOI: 10.1109/TAC.2016.2613905.

- [27] Phan, Minh, Horta, Lucas G, Juang, Jer-Nan, and Longman, Richard W. Linear system identification via an asymptotically stable observer. *Journal of Optimization Theory and Applications*, 79(1):59–86, 1993. DOI: 10.1007/bf00941887.
- [28] Proskurnikov, Anton V. and Tempo, Roberto. A tutorial on modeling and analysis of dynamic social networks. Part I. *Annual Reviews in Control*, 43:65–79, 2017. DOI: 10.1016/j.arcontrol.2017.03.002.
- [29] Proskurnikov, Anton V. and Tempo, Roberto. A tutorial on modeling and analysis of dynamic social networks. Part II. *Annual Reviews in Control*, 45:166–190, 2018. DOI: 10.1016/j.arcontrol.2018.03.005.
- [30] Qu, Jijun, Ji, Zhijian, Lin, Chong, and Yu, Haisheng. Fast consensus seeking on networks with antagonistic interactions. *Complexity*, 2018, 2018. DOI: 10.1155/2018/7831317.
- [31] Roweis, Sam and Ghahramani, Zoubin. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–345, 1999. DOI: 10.1162/089976699300016674.
- [32] Shields, R. and Pearson, J. Structural controllability of multiinput linear systems. *IEEE Transactions on Automatic Control*, 21(2):203–212, 1976. DOI: 10.1109/tac.1976.1101198.
- [33] Siljak, Dragoslav D. *Decentralized control of complex systems*. Courier Corporation, 2011.
- [34] Songsiri, Jitkomut and Vandenberghe, Lieven. Topology selection in graphical models of autoregressive processes. *The Journal of Machine Learning Research*, 11:2671–2705, 2010.
- [35] Sun, Yinshuang, Ji, Zhijian, and Liu, Kaien. Event-based consensus for general linear multiagent systems under switching topologies. *Complexity*, 2020, 2020. DOI: 10.1155/2020/5972749.
- [36] Villaverde, Alejandro F. and Barreiro, Antonio. Identifiability of large nonlinear biochemical networks. *MATCH Commun. Math. Comput. Chem.*, 76:259–296, 2016.
- [37] Walter, Eric and Pronzato, Luc. *Identification of parametric models: from experimental data*. Springer Verlag, 1997.
- [38] Wolkenhauer, Olaf, Wellstead, Peter, Cho, Kwang-Hyun, and Sontag, Eduardo D. Network reconstruction based on steady-state data. *Essays in Biochemistry*, 45:161–176, 2008. DOI: 10.1042/bse0450161.
- [39] Zachary, Wayne W. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473, 1977. DOI: 10.1086/jar.33.4.3629752.

Appendix A: Proof of Proposition 5

Proof. Since A^d returned by **FindDenseRealization** is a convex combination of dynamical equivalent realizations computed by Algorithm **FindRealization**, $A^d \in \mathcal{A}^p(\mathcal{Y}, B, C, D)$ holds and $[A^d]_{ij} = 0$ for all $(i, j) \in L$.

Assume that A^d returned by **FindDenseRealization** is not dense in $\mathcal{A}^p(\mathcal{Y}, B, C, D)$ among the state transition matrices satisfying the zero-constraints defined by L . Then there exists a tuple (i, j) , $i \neq j$ for which there is a realization $A' \in \mathcal{A}_M^p$ so that $[A']_{ij} > 0$, but $[A^d]_{ij} = 0$. By construction it is guaranteed that Algorithm **FindDenseRealization** has at least one iteration in which the optimization objective to be maximized involves the entry indexed by (i, j) , i.e. $(i, j) \in H$. Then it follows that $[A^d]_{ij} > 0$ which is a contradiction.

Since **FindDenseRealization** computes a linear program of the form E.q. 22 at most $(n^2 - n)$ -times, A^d is obtained in polynomial time. □

Appendix B: Proof of Proposition 6

Proof. Let us assume that there exists a sequence $V \in \mathcal{R}$ encoding a feasible state transition matrix which is not returned by the algorithm **FindAllRealizations**. Then consider the sequence R for which $R =_p V$ and p is maximal. If $p = 0$ then the encoding sequence of the dynamically equivalent dense realization is an appropriate choice for R . For $i = \mathbf{FindNextOne}(R, p)$ and $j = \mathbf{FindNextOne}(V, p)$ we have $i \leq j$, since $V \in C_p(R)$. Moreover if $i = j$ then it follows that p is not the maximal integer such that $R =_p V$ which is a contradiction.

Let us consider the sequence W_1 returned by **FindReal**(R, p, i). There exists a dynamically equivalent realization encoded by W_1 , since the input constraints of **FindReal**(R, p, i) are fulfilled by V . For W_1 we get that $j_1 = \mathbf{FindNextOne}(W_1, p)$ for some $j_1 \in \mathbb{Z}_{>0}$. Then the inequality $j_1 \leq j$ must hold, since $V \in C_i(W_1)$. If $j_1 = j$ then it would follow that R is not that sequence for which $V \in C_p(R)$ holds with a maximal p , i.e. $j_1 = j$ is a contradiction.

There is a step in the algorithm **FindAllRealizations** when the sequence W_2 is computed by **FindDenseRealizationSequence**($\Theta = (A, B, C, D), R, p, j_1$). For W_2 we have that $V \in C_{j_1}(W_2)$ which implies that $j_2 \leq j$ for $j_2 = \mathbf{FindNextOne}(R, p, j_1)$. If $j = j_2$ would hold, then j_2 would be the maximal integer with the sequence W_2 for which $W_2 =_{j_2} V$ holds, but this is a contradiction.

Continuing the above steps would lead to infinitely many valid graph structures which is a contradiction. □

Appendix C: Adjacency matrix of Example 2

Non-zero entries in the initial adjacency matrix of Example 2.

$$\begin{aligned}
& [A]_{2,10} = 1; \\
& [A]_{3,1} = 0.5; [A]_{3,2} = 0.5; \\
& [A]_{4,1} = 0.3; [A]_{4,2} = 0.5; [A]_{4,3} = 0.2; \\
& [A]_{5,1} = 1; \\
& [A]_{6,1} = 1; \\
& [A]_{7,1} = 0.3; [A]_{7,5} = 0.4; [A]_{7,6} = 0.3; \\
& [A]_{8,1} = 0.2; [A]_{8,2} = 0.2; [A]_{8,3} = 0.2; [A]_{8,4} = 0.4; \\
& [A]_{9,1} = 0.4; [A]_{9,3} = 0.6; \\
& [A]_{10,3} = 1.0; \\
& [A]_{11,1} = 0.1; [A]_{11,5} = 0.1; [A]_{11,6} = 0.8; \\
& [A]_{12,1} = 1.0; \\
& [A]_{13,1} = 0.9; [A]_{13,4} = 0.1; \\
& [A]_{14,1} = 0.1; [A]_{14,2} = 0.2; [A]_{14,3} = 0.3; [A]_{14,4} = 0.4; \\
& [A]_{17,6} = 0.7; [A]_{17,7} = 0.3; \\
& [A]_{18,1} = 0.8; [A]_{18,2} = 0.2; \\
& [A]_{20,1} = 0.1; [A]_{20,2} = 0.9; \\
& [A]_{22,1} = 0.4; [A]_{22,2} = 0.6; \\
& [A]_{26,24} = 0.65; [A]_{26,25} = 0.35; \\
& [A]_{28,3} = 0.2; [A]_{28,24} = 0.3; [A]_{28,25} = 0.5; \\
& [A]_{29,3} = 1.0; \\
& [A]_{30,24} = 0.3; [A]_{30,27} = 0.7; \\
& [A]_{31,2} = 0.1; [A]_{31,9} = 0.9; \\
& [A]_{32,1} = 0.25; [A]_{32,25} = 0.25; [A]_{32,26} = 0.4; [A]_{32,29} = 0.1; \\
& [A]_{33,3} = 0.1; [A]_{33,9} = 0.1; [A]_{33,15} = 0.1; [A]_{33,16} = 0.1; [A]_{33,19} = 0.05; [A]_{33,21} = 0.05; \\
& [A]_{33,23} = 0.1; [A]_{33,24} = 0.1; [A]_{33,30} = 0.1; [A]_{33,31} = 0.1; [A]_{33,32} = 0.1; \\
& [A]_{34,9} = 0.05; [A]_{34,10} = 0.05; [A]_{34,14} = 0.15; [A]_{34,15} = 0.05; [A]_{34,16} = 0.01; \\
& [A]_{34,19} = 0.09; [A]_{34,20} = 0.02; [A]_{34,21} = 0.08; [A]_{34,23} = 0.03; [A]_{34,24} = 0.07; \\
& [A]_{34,27} = 0.1; [A]_{34,28} = 0.05; [A]_{34,29} = 0.05; [A]_{34,30} = 0.05; [A]_{34,31} = 0.05; \\
& [A]_{34,32} = 0.05; [A]_{34,33} = 0.05
\end{aligned}$$

Received 28th December 2020

The Inverse Epsilon Distribution as an Alternative to Inverse Exponential Distribution with a Survival Times Data Example*

Tamás Jónás^a, Christophe Chesneau^b, József Dombi^c, and Hassan S. Bakouch^d

Abstract

This paper is devoted to a new flexible two-parameter lower-truncated distribution, which is based on the inversion of the so-called epsilon distribution. It is called the inverse epsilon distribution. In some senses, it can be viewed as an alternative to the inverse exponential distribution, which has many applications in reliability theory and biology. Diverse properties of the new lower-truncated distribution are derived including relations with existing distributions, hazard and reliability functions, survival and reverse hazard rate functions, stochastic ordering, quantile function with related skewness and kurtosis measures, and moments. A demonstrative survival times data example is used to show the applicability of the new model.

Keywords: epsilon distribution, inverse exponential distribution, inverse epsilon distribution

1 Introduction

The exponential distribution and its generalizations play an important role in many areas of science, including physics, chemistry, medical sciences and reliability engineering (see e.g. [1, 2, 16, 19]). Dombi et al. [6] introduced the epsilon distribution,

*This research was partially supported by the project ‘Integrated program for training new generation of scientists in the fields of computer science’, no EFOP- 3.6.3-VEKOP-16-2017-0002. The project has been supported by the European Union and co-funded by the European Social Fund.

^aFaculty of Economics, Eötvös Loránd University, Rákóczi út 7, H-1088 Budapest, Hungary, E-mail: jonas@gtk.elte.hu, ORCID: 0000-0001-8241-2321

^bLaboratoire de Mathématiques Nicolas Oresme, University of Caen Normandie, Caen, France, E-mail: christophe.chesneau@unicaen.fr, ORCID: 0000-0002-1522-9292

^cDepartment of Computer Algorithms and Artificial Intelligence, University of Szeged, Árpád tér 2, H-6720 Szeged, Hungary, E-mail: dombi@inf.u-szeged.hu, ORCID: 0000-0001-9459-912X

^dDepartment of Mathematics, Faculty of Science, Tanta University Al-Geish St., Tanta, Gharbia, Egypt, E-mail: hassan.bakouch@science.tanta.edu.eg, ORCID: 0000-0002-3189-0670

which may be treated as an alternative to the exponential distribution. Here, we will briefly review the epsilon distribution and its connection with the exponential distribution. Dombi et al. [6] defined the epsilon function as follows.

Definition 1. The epsilon function $\varepsilon_{\lambda,d}(x): (-d, d) \rightarrow (0, \infty)$ is given by

$$\varepsilon_{\lambda,d}(x) = \left(\frac{d+x}{d-x} \right)^{\lambda \frac{d}{2}},$$

where $\lambda \in \mathbb{R}$, $\lambda \neq 0$, $d \in \mathbb{R}$, $d > 0$.

The following proposition concerns a key property of the epsilon function.

Proposition 1. For any $x \in (-d, +d)$, if $d \rightarrow \infty$, then

$$\varepsilon_{\lambda,d}(x) \rightarrow e^{\lambda x}.$$

Proof. See the proof of Theorem 1 in [6]. □

Utilizing the epsilon function given in Definition 1, the continuous random variable X said to have an epsilon distribution with the parameters $\lambda > 0$ and $d > 0$, if its cumulative distribution function (CDF) is given by

$$F_{\lambda,d}(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ 1 - \varepsilon_{-\lambda,d}(x), & \text{if } 0 < x < d \\ 1, & \text{if } x \geq d. \end{cases}$$

Notation 1. From now on, $X \sim \varepsilon(\lambda, d)$ will denote that the random variable X has an epsilon distribution with the parameters $\lambda > 0$ and $d > 0$.

Exploiting Proposition 1, we can state the following proposition.

Proposition 2. Let $X \sim \varepsilon(\lambda, d)$ and let $Y \sim \exp(\lambda)$, where $\lambda > 0, d > 0$. Then, for any $x \in \mathbb{R}$

$$\lim_{d \rightarrow \infty} P(X < x) = P(Y < x).$$

Proof. By making use of the definitions for the epsilon and the exponential distributions, the proposition immediately follows from Proposition 1. □

Based on Proposition 2, we may state that the asymptotic epsilon distribution is just the exponential distribution. It is worth mentioning that while the hazard function of an exponentially distributed random variable is constant, the hazard function of a random variable with an epsilon distribution can exhibit both constant and increasing shapes. That is, in reliability analyses, the epsilon distribution can be utilized to describe the distribution of the time to first failure random variable both in the second and in the third phases of the hazard function.

The reciprocal of a random variable with an exponential distribution is said to be a random variable with an inverse exponential distribution. The inverse exponential

distribution, like the exponential distribution, has a wide range of applications (see e.g. [17]). For example, if a random variable with an exponential distribution represents the time between failures of a system, then the reciprocal of this random variable, which has an inverse exponential distribution, describes the frequency of the system failures over time.

In this study, we will present the inverse epsilon distribution and, by the means of an illustrative data example, show that it may be viewed as an alternative to the inverse exponential distribution. The key features of the inverse epsilon distribution and the main motivations of our study can be summarized as follows:

- (a) It is a new, flexible, lower-truncated power-polynomial distribution.
- (b) The famous inverse exponential distribution is just the limit of the inverse epsilon distribution.
- (c) The literature lacks of a flexible inverted lower-truncated distributions.
- (d) The hazard function of the inverse exponential distribution has a first, increasing part and a second, slowly decreasing part (see [18]). This explains why in the course of the study of mortality associated with some diseases, the inverse exponential distribution may be utilized as a life distribution model (see [12, 5]). Taking into account the asymptotic property of the inverse epsilon distribution, this latter one can also be utilized in mortality studies.

This paper is structured as follows. In Section 2, we will introduce the inverse epsilon distribution and describe its key properties including the hazard function, survival and reverse hazard rate functions, stochastic ordering, quantile function and moments. Next, in Section 3, we will present a demonstrative example of the application of the new distribution on survival times data. Lastly, in Section 4, our main findings are summarized.

2 Theoretical aspects

2.1 Basics on the inverse epsilon distribution

Here, we will present the inverse epsilon distribution and show that it may be viewed as an alternative to the inverse exponential distribution.

Now, let the random variable X have an epsilon distribution with the parameters $\lambda, d > 0$; that is, $X \sim \varepsilon(\lambda, d)$. Next, let $Y = 1/X$, where $X > 0$, and let $G_{\lambda,d}: \mathbb{R}^+ \rightarrow (0, 1)$ be the CDF of Y . Then, noting that X and Y are continuous random variables, after direct calculation, we have

$$G_{\lambda,d}(y) = P(Y < y) = P\left(X > \frac{1}{y}\right) = 1 - P\left(X \leq \frac{1}{y}\right) = 1 - F_{\lambda,d}\left(\frac{1}{y}\right).$$

Therefore,

$$G_{\lambda,d}(y) = \begin{cases} 0, & \text{if } 0 < y \leq \frac{1}{d} \\ \left(\frac{d+y^{-1}}{d-y^{-1}}\right)^{-\lambda \frac{d}{2}}, & \text{if } y > \frac{1}{d}. \end{cases}$$

By taking the derivative of function $G_{\lambda,d}(x)$, we get the probability density function (PDF) $g_{\lambda,d}(x) = G'_{\lambda,d}(x)$ of the random variable Y :

$$g_{\lambda,d}(y) = \begin{cases} 0, & \text{if } 0 < y \leq \frac{1}{d} \\ \lambda \frac{d^2}{d^2 y^2 - 1} \left(\frac{d+y^{-1}}{d-y^{-1}} \right)^{-\lambda \frac{d}{2}}, & \text{if } y > \frac{1}{d}. \end{cases}$$

Following this line of thinking, we define the inverse epsilon distribution as follows.

Definition 2. *The continuous random variable $X > 0$ has an inverse epsilon distribution with the parameters $\lambda > 0$ and $d > 0$, if the PDF $f_{\lambda,d}$ of X is given by*

$$f_{\lambda,d}(x) = \begin{cases} 0, & \text{if } 0 < x \leq \frac{1}{d} \\ \lambda \frac{d^2}{d^2 x^2 - 1} \left(\frac{d+x^{-1}}{d-x^{-1}} \right)^{-\lambda \frac{d}{2}}, & \text{if } x > \frac{1}{d}. \end{cases} \quad (1)$$

Note that the CDF of the inverse epsilon distribution given in Definition 2 is

$$F_{\lambda,d}(x) = \begin{cases} 0, & \text{if } 0 < x \leq \frac{1}{d} \\ \left(\frac{d+x^{-1}}{d-x^{-1}} \right)^{-\lambda \frac{d}{2}}, & \text{if } x > \frac{1}{d}. \end{cases} \quad (2)$$

Notation 2. *From now on, $X \sim \bar{\varepsilon}(\lambda, d)$ will denote that the random variable $X > 0$ has an inverse epsilon distribution with the parameters $\lambda > 0$ and $d > 0$.*

It is a familiar fact that a continuous random variable $X > 0$ has an inverse exponential distribution with the parameter $\lambda > 0$, if the PDF $f_{\lambda}(x)$ and the CDF $F_{\lambda}(x)$ of X are given by

$$f_{\lambda}(x) = \frac{\lambda}{x^2} e^{-\lambda \frac{1}{x}}, \quad F_{\lambda}(x) = e^{-\lambda \frac{1}{x}}, \quad (3)$$

respectively.

Notation 3. *Hereafter, $X \sim \text{invexp}(\lambda)$ will denote that the random variable $X > 0$ has an inverse exponential distribution with the parameter $\lambda > 0$.*

The following proposition concerns the connection between the inverse exponential and inverse epsilon distributions.

Proposition 3. *Let $X \sim \bar{\varepsilon}(\lambda, d)$ and let $Y \sim \text{invexp}(\lambda)$, where $\lambda > 0, d > 0$ and $X, Y > 0$. Then, for any $x > 0$*

$$\lim_{d \rightarrow \infty} P(X < x) = P(Y < x).$$

Proof. Noting the CDFs of X and Y given in Eq. (2) and Eq. (3), respectively, and applying Proposition 1, for any $x > 0$, we can write

$$\lim_{d \rightarrow \infty} P(X < x) = \lim_{d \rightarrow \infty} \left(\frac{d+x^{-1}}{d-x^{-1}} \right)^{-\lambda \frac{d}{2}} = e^{-\lambda \frac{1}{x}} = P(Y < x).$$

□

Based on Proposition 3, the inverse exponential distribution may be viewed as the asymptotic inverse epsilon distribution.

Using the results above, we may state that the interests in the inverse epsilon distribution is based on the following facts:

- (a) It is a new, flexible, lower-truncated power-polynomial distribution.
- (b) The famous inverse exponential distribution is just the limit of the inverse epsilon distribution.
- (c) The literature lacks of a flexible inverted lower-truncated distributions.

Some example plots of the CDFs of the inverse epsilon distribution are shown in Figure 1.

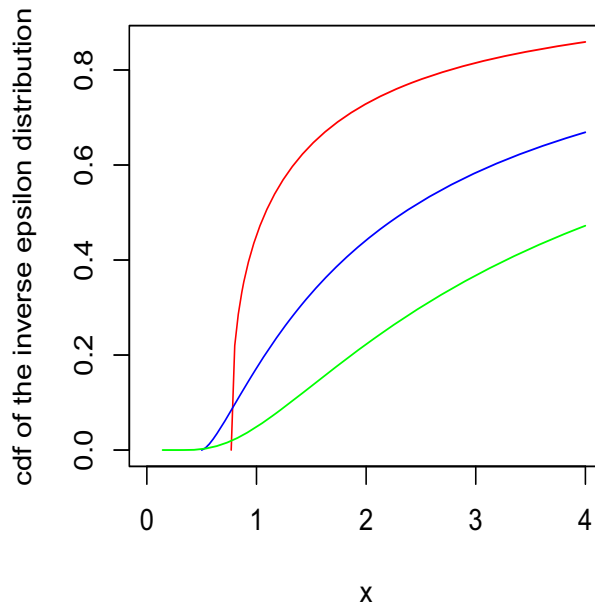


Figure 1: CDF of the inverse epsilon distribution with three sets of parameters for (λ, d) : $(0.6, 1.3)$, $(1.6, 2)$ and $(3, 7)$.

We observe that the CDF can be more or less concave (for the considered values).

2.2 Hazard function

By making use of the PDF and CDF of the inverse exponential distribution, we get that its hazard function $h_\lambda: (0, \infty) \rightarrow (0, \infty)$ is

$$h_\lambda(x) = \frac{f_\lambda(x)}{1 - F_\lambda(x)} = \frac{\frac{\lambda}{x^2} e^{-\lambda \frac{1}{x}}}{1 - e^{-\lambda \frac{1}{x}}},$$

where $\lambda > 0$.

Using the PDF and the CDF of the inverse epsilon distribution with the parameters $\lambda, d > 0$, the hazard function $h_{\lambda,d}: (0, \infty) \rightarrow [0, \infty)$ of this distribution is

$$h_{\lambda,d}(x) = \frac{f_{\lambda,d}(x)}{1 - F_{\lambda,d}(x)} = \begin{cases} 0, & \text{if } 0 < x \leq \frac{1}{d} \\ \lambda \frac{\frac{d^2}{d^2 x^2 - 1} \left(\frac{d+x-1}{d-x-1}\right)^{-\lambda \frac{d}{2}}}{1 - \left(\frac{d+x-1}{d-x-1}\right)^{-\lambda \frac{d}{2}}}, & \text{if } x > \frac{1}{d}. \end{cases}$$

Proposition 4. *Let $X \sim \bar{\varepsilon}(\lambda, d)$ and let $Y \sim \text{invexp}(\lambda)$, where $\lambda > 0, d > 0$ and $X, Y > 0$. Furthermore let $h_{\lambda,d}: (0, \infty) \rightarrow [0, \infty)$ and $h_\lambda: (0, \infty) \rightarrow (0, \infty)$ be the hazard functions of X and Y , respectively. Then, for any $x > 0$*

$$\lim_{d \rightarrow \infty} h_{\lambda,d}(x) = h_\lambda(x).$$

Proof. This proposition immediately follows from Proposition 1. □

The first derivative of the hazard function $h_{\lambda,d}(x)$ is

$$\frac{dh_{\lambda,d}(x)}{dx} = - \frac{\lambda d^4 \left((2x - \lambda) \left(\frac{dx+1}{dx-1}\right)^{\frac{\lambda d}{2}} - 2x \right)}{(dx - 1)^2 (dx + 1)^2 \left(\left(\frac{dx+1}{dx-1}\right)^{\frac{\lambda d}{2}} - 1 \right)^2}.$$

Using the first derivative of $h_{\lambda,d}(x)$, one can see that

- if $0 < \lambda d \leq 2$, then $h_{\lambda,d}(x)$ is strictly decreasing in the interval $(\frac{1}{d}, \infty)$
- if $\lambda d > 2$, then in the interval $(\frac{1}{d}, \infty)$, $h_{\lambda,d}(x)$ is first increasing, and then decreasing; that is, $h_{\lambda,d}(x)$ has a local maxima.

Figure 2 shows example plots of hazard functions of the inverse exponential distribution and the inverse epsilon distribution.

It is an acknowledged fact that in the course of the study of mortality associated with some diseases, the hazard function has a first, increasing part and a second, slowly decreasing part [18]. We can see that the hazard function of the inverse exponential distribution (see the left upper plot in Figure 2) exhibits such a shape. This is why the inverse exponential distribution may be utilized as a life distribution model (see [12, 5]).

Now, by taking into account the above mentioned characteristics of the hazard function of the inverse epsilon distribution, we can draw the following practical conclusions.

- The hazard function of the inverse epsilon distribution with the parameters $\lambda, d > 0$ may be viewed as an alternative to the hazard function of the inverse exponential distribution, if the value of parameter d is sufficiently large. If $d \rightarrow \infty$, then the two hazard functions coincide.

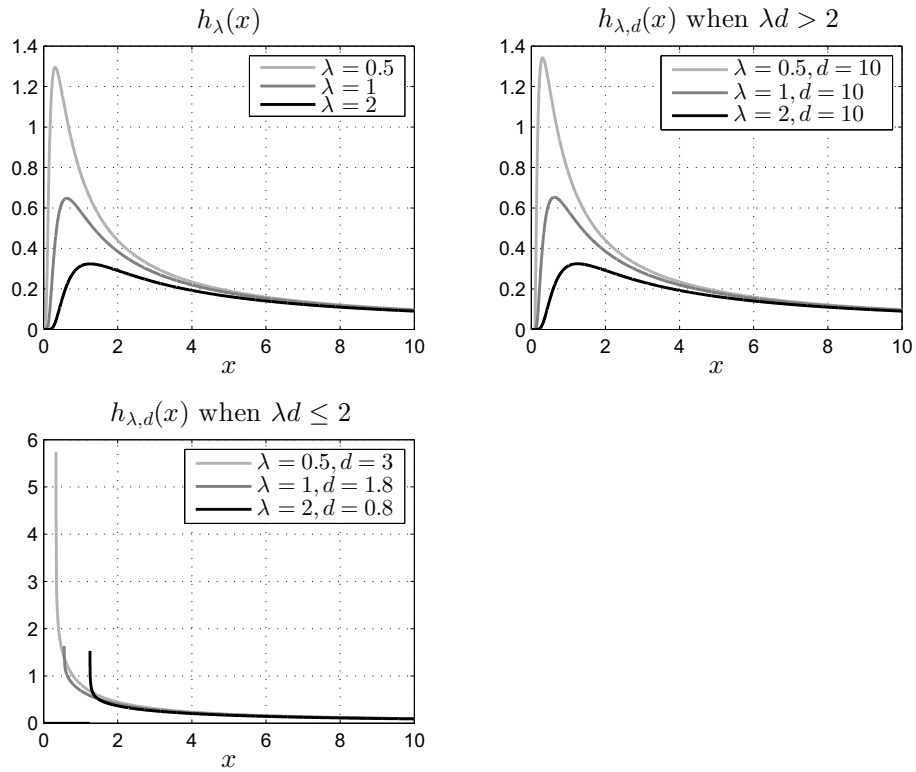


Figure 2: Example plots of hazard functions of the inverse exponential distribution and the inverse epsilon distribution.

- If $\lambda d > 2$, then the shape of the hazard function of the inverse epsilon distribution is very similar to that of the hazard function of the inverse exponential distribution. In this case, the hazard function is first increasing and then it is slowly decreasing (see the upper plots in Figure 2).
- If $0 < \lambda d \leq 2$, then the hazard function of the inverse epsilon distribution is strictly decreasing in the interval $(\frac{1}{d}, \infty)$ (see the lower plot in Figure 2).

Therefore, the inverse epsilon distribution can be used to model life time data that have either first monotonically increasing and then decreasing hazard rates, or monotonically decreasing hazard rates.

2.3 Survival and reverse hazard rate functions

The following functions are of interest, mainly in hazard and reliability analysis. The survival function of the inverse epsilon distribution is obtained as

$$S_{\lambda,d}(x) = 1 - F_{\lambda,d}(x) = \begin{cases} 1, & \text{if } 0 < x \leq \frac{1}{d} \\ 1 - \left(\frac{d+x^{-1}}{d-x^{-1}}\right)^{-\lambda\frac{d}{2}}, & \text{if } x > \frac{1}{d}. \end{cases}$$

The reversed hazard rate function of the inverse epsilon distribution is given by

$$r_{\lambda,d}(x) = \frac{f_{\lambda,d}(x)}{F_{\lambda,d}(x)} = \begin{cases} 0, & \text{if } 0 < x \leq \frac{1}{d} \\ \lambda \frac{d^2}{d^2 x^2 - 1}, & \text{if } x > \frac{1}{d}. \end{cases}$$

The cumulative hazard rate function is expressed as

$$H_{\lambda,d}(x) = -\ln(S_{\lambda,d}(x)) = \begin{cases} 0, & \text{if } 0 < x \leq \frac{1}{d} \\ -\ln \left[1 - \left(\frac{d+x^{-1}}{d-x^{-1}}\right)^{-\lambda\frac{d}{2}} \right], & \text{if } x > \frac{1}{d}, \end{cases}$$

where the logarithmic term can be decomposed as

$$\begin{aligned} -\ln \left[1 - \left(\frac{d+x^{-1}}{d-x^{-1}}\right)^{-\lambda\frac{d}{2}} \right] &= -\ln \left[(d-x^{-1})^{-\lambda\frac{d}{2}} - (d+x^{-1})^{-\lambda\frac{d}{2}} \right] \\ &\quad - \lambda\frac{d}{2} \ln(d-x^{-1}). \end{aligned}$$

Further details on these functions can be found in [11].

2.4 Stochastic ordering

The following stochastic ordering result on the inverse epsilon distribution holds.

Proposition 5. *Let $F_{\lambda,d}$ be the CDF of the inverse epsilon distribution as defined by (2). Then, for $d_2 \geq d_1 > 0$ and any $x > 0$, we have*

$$F_{\lambda,d_2}(x) \geq F_{\lambda,d_1}(x);$$

and for any $\lambda_2 \geq \lambda_1 > 0$ and any $x > 0$, we have

$$F_{\lambda_1,d}(x) \geq F_{\lambda_2,d}(x).$$

Proof. For $x < 1/d_2$, the CDFs are equal to 0. For $x \in [1/d_2, 1/d_1]$, since $F_{\lambda,d_1}(x) = 0$, the inequality is clear too. Now, for $x > 1/d_1 > 1/d_2$, by using the following inequality:

$$\frac{1}{2} \ln \left(\frac{1+x}{x-1} \right) < \frac{x}{x^2-1}$$

for $x > 1$, we get

$$\frac{\partial}{\partial d} F_{\lambda,d}(x) = \lambda \left(\frac{d+x^{-1}}{d-x^{-1}} \right)^{-\lambda \frac{d}{2}} \left[\frac{dx}{d^2 x^2 - 1} - \frac{1}{2} \ln \left(\frac{1+dx}{dx-1} \right) \right] > 0,$$

implying that $F_{\lambda,d}(x)$ is increasing with respect to d .

For $\lambda_2 \geq \lambda_1$, we have

$$F_{\lambda_1,d}(x) \geq F_{\lambda_2,d}(x).$$

Indeed, the function $F_{\lambda,d}(x)$ is decreasing with respect to λ : we have

$$\frac{\partial}{\partial \lambda} F_{\lambda,d}(x) = -\frac{d}{2} \left(\frac{d+x^{-1}}{d-x^{-1}} \right)^{-\lambda \frac{d}{2}} \ln \left(\frac{d+x^{-1}}{d-x^{-1}} \right) < 0.$$

□

Under the conditions of Proposition 5, we see that $X_1 \sim \bar{\varepsilon}(\lambda_1, d)$ first order stochastically dominates $X_2 \sim \bar{\varepsilon}(\lambda_2, d)$.

2.5 Quantile function

The quantile function of the inverse epsilon distribution is obtained by inverting $F_{\lambda,d}(x)$. After some developments, we arrive at

$$Q_{\lambda,d}(u) = \frac{1}{d} \left(\frac{u^{-\frac{2}{d\lambda}} + 1}{u^{-\frac{2}{d\lambda}} - 1} \right) = \frac{1}{d} \left(\frac{1 + u^{\frac{2}{d\lambda}}}{1 - u^{\frac{2}{d\lambda}}} \right), \quad u \in (0, 1).$$

This function is of importance because it allows us to define the main quartiles of the inverse epsilon distribution, as the first quartile: $Q_{\lambda,d}(1/4)$, the median: $Q_{\lambda,d}(1/2)$ and the third quartile: $Q_{\lambda,d}(3/4)$. Also, it can be served to generate values from the inverse epsilon distribution.

We should also add that we can use $Q_{\lambda,d}(u)$ to define measures of skewness and kurtosis as the Bowley skewness and Moors kurtosis are given by

$$B_{\lambda,d} = \frac{Q_{\lambda,d}(1/4) - 2Q_{\lambda,d}(1/2) + Q_{\lambda,d}(3/4)}{Q_{\lambda,d}(3/4) - Q_{\lambda,d}(1/4)}$$

and

$$M_{\lambda,d} = \frac{Q_{\lambda,d}(7/8) - Q_{\lambda,d}(5/8) + Q_{\lambda,d}(3/8) - Q_{\lambda,d}(1/8)}{Q_{\lambda,d}(6/8) - Q_{\lambda,d}(2/8)}.$$

These measures provide alternative definitions to the skewness and kurtosis measures defined with moments. For more details on these alternative definitions see [8] and [13].

Figure 3 shows the plots of Bowley skewness and Moors kurtosis as functions of the parameters λ and d . The graphics for $B_{\lambda,d}$ and $M_{\lambda,d}$ are useful to determine

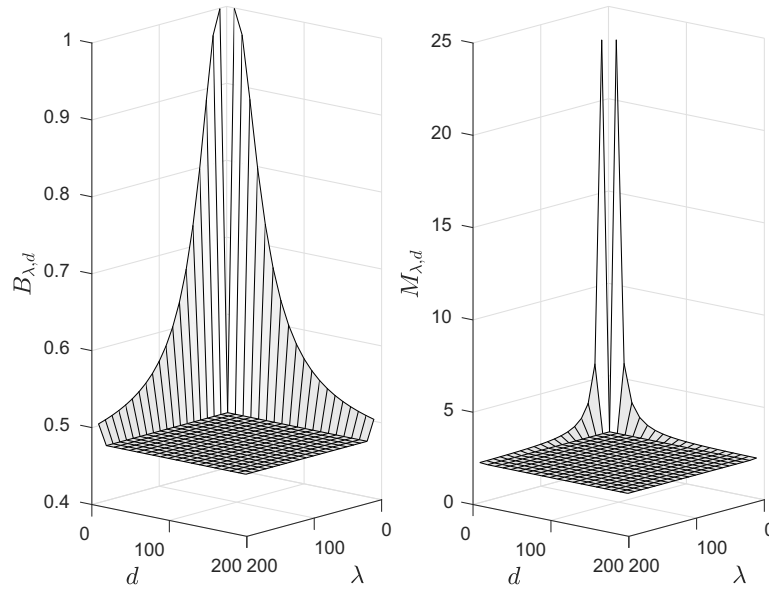


Figure 3: Plots of Bowley skewness and Moors kurtosis

the ability of the inverse epsilon distribution in skewness and kurtosis. This is very interesting for the inverse epsilon distribution because it does not admit mean (and obviously raw moments of superior order). This aspect is developed in the next section.

Also, upon differentiation of $Q_{\lambda,d}(u)$ according to u , the quantile density function is defined by

$$q_{\lambda,d}(u) = \frac{4}{\lambda d^2} \frac{u^{\frac{2}{d\lambda}-1}}{\left(1 - u^{\frac{2}{d\lambda}}\right)^2}, \quad u \in (0, 1).$$

This function is of interest since it appears in several statistical tools. For further details see [10].

2.6 Moments

Let us now investigate the moments of $X \sim \bar{\varepsilon}(\lambda, d)$. Then, assuming that it exists, the mean of X^r is defined by

$$\mu'_r = E(X^r) = \int_{1/d}^{+\infty} x^r f_{\lambda,d}(x) dx = \int_{1/d}^{+\infty} x^r \lambda \frac{d^2}{d^2 x^2 - 1} \left(\frac{d+x^{-1}}{d-x^{-1}}\right)^{-\lambda \frac{d}{2}} dx.$$

Proposition 6. *The mean of X^r exists if and only if $r \in (-\lambda d/2, 1)$, and it is given as*

$$\mu'_r = \frac{\lambda}{d^r} B\left(1-r, \frac{\lambda d}{2} + 1\right) {}_2F_1\left(\frac{\lambda d}{2} + 1, 1-r; 1-r + \frac{\lambda d}{2}; -1\right),$$

where $B(a, b)$ and ${}_2F_1(a, b; c; x)$ are the classical beta and Gauss hypergeometric functions, respectively.

Proof. When $x \rightarrow 1/d$, we have $f_{\lambda, d}(x) \sim \lambda d^2 2^{-\lambda \frac{d}{2}-1} (xd - 1)^{\lambda \frac{d}{2}-1}$ so, by the Riemann integrability, the integral converge in 0 if and only if $1 - \lambda d/2 - r < 1$, hence $r > -\lambda d/2$. Also, when $x \rightarrow +\infty$, we have $f_{\lambda, d}(x) \sim \lambda x^{r-2}$ so, by the Riemann integrability, the integral converge in $+\infty$ if and only if $2 - r > 1$, hence $r < 1$. That is, μ'_r exists if and only if $r \in (-\lambda d/2, 1)$. Following the lines of [15] with the use of $Y \sim \bar{\varepsilon}(\lambda, d)$ and the change of variable $y = x/d$, we have

$$\begin{aligned} \mu'_r &= E(Y^{-r}) = \lambda d^2 \int_0^d \frac{x^{-r}}{d^2 - x^2} \left(\frac{d+x}{d-x}\right)^{-\lambda \frac{d}{2}} dx \\ &= \frac{\lambda}{d^r} \int_0^1 y^{-r} (1-y)^{\lambda \frac{d}{2}-1} (1+y)^{-\lambda \frac{d}{2}-1} dy. \end{aligned}$$

The desired result involving the beta and Gauss hypergeometric functions is an immediate application of Eq. 3.197.3 of [9]. □

In particular, from Proposition 6, we see that the mean of X doesn't exist. Some inverse raw moments of X exists, depending on the large values for λ and d .

Remark 1. Alternatively, by applying the change of variable $x = Q_{\lambda, d}(u)$ and use he general binomial theorem, one can also express μ'_r as

$$\begin{aligned} \mu'_r &= \int_0^1 [Q_{\lambda, d}(u)]^r du = \frac{1}{d^r} \int_0^1 \left(\frac{1+u \frac{2}{d\lambda}}{1-u \frac{2}{d\lambda}}\right)^r du \\ &= \frac{1}{d^r} \int_0^1 \left[\sum_{k=0}^{+\infty} \binom{r}{k} u^{k \frac{2}{d\lambda}} \right] \left[\sum_{\ell=0}^{+\infty} \binom{-r}{\ell} (-1)^\ell u^{\ell \frac{2}{d\lambda}} \right] du \\ &= \frac{1}{d^r} \sum_{k, \ell=0}^{+\infty} \binom{r}{k} \binom{-r}{\ell} (-1)^\ell \frac{1}{2(k+\ell)/(d\lambda) + 1}, \end{aligned}$$

from which an acceptable approximation can be given by substituting $+\infty$ by any large integer. If λ or d are sufficiently large, the negative moments of X can be investigated for moments analysis.

The incomplete moments of X exists when $r \geq 0$; the r th incomplete moment of X^r at $t > 1/d$ is given by

$$\mu'_r(t) = E(X^r 1_{\{X \leq t\}}) = \int_{1/d}^t x^r f_{\lambda, d}(x) dx = \int_{1/d}^t x^r \lambda \frac{d^2}{d^2 x^2 - 1} \left(\frac{d+x^{-1}}{d-x^{-1}}\right)^{-\lambda \frac{d}{2}} dx$$

or, equivalently,

$$\mu'_r(t) = \int_0^{F_{\lambda,d}(t)} [Q_{\lambda,d}(u)]^r du = \frac{1}{d^r} \int_0^{\left(\frac{d+t-1}{d-t-1}\right)^{-\lambda \frac{d}{2}}} \left(\frac{1+u \frac{2}{d\lambda}}{1-u \frac{2}{d\lambda}}\right)^r du.$$

To our knowledge, there is no close form $\mu'_r(t)$. For known parameters (including t), we can have a numerical value of it. As a complementary approach, a series expansion of $\mu'_r(t)$ is possible through the application of the generalized binomial series expansion. Following this approach, we get

$$\mu'_r(t) = \frac{1}{d^r} \sum_{k,\ell=0}^{+\infty} \binom{r}{k} \binom{-r}{\ell} (-1)^\ell \frac{1}{2(k+\ell)/(d\lambda) + 1} \left(\frac{d+t-1}{d-t-1}\right)^{-(k+\ell)-\lambda \frac{d}{2}}.$$

From the incomplete moments, one can define applied curves, functions or indexes of interest, such as the Lorenz curves, Gini index and mean residual life or others. See, for instance, [3].

3 A demonstrative survival times data example

Oguntunde et al. [14] used the following data of survival times (in days) of a group of patients suffering from head and neck cancer diseases and treated using a combination of radiotherapy and chemotherapy (see [7]):

12.20, 23.56, 23.74, 25.87, 31.98, 37, 41.35, 47.38, 55.46, 58.36, 63.47, 68.46, 78.26, 74.47, 81.43, 84, 92, 94, 110, 112, 119, 127, 130, 133, 140, 146, 155, 159, 173, 179, 194, 195, 209, 249, 281, 319, 339, 432, 469, 519, 633, 725, 817, 1776.

Oguntunde et al. [14] modeled these survival times using the exponential inverse exponential (EIE) distribution that has the following PDF and CDF, respectively,

$$f_{\theta,\alpha}(x) = \alpha \frac{\theta}{x^2} e^{-\frac{\theta}{x}} \frac{1}{\left(1 + e^{-\frac{\theta}{x}}\right)^2} e^{-\alpha \frac{e^{-\frac{\theta}{x}}}{1 - e^{-\frac{\theta}{x}}}} \quad (4)$$

$$F_{\theta,\alpha}(x) = 1 - e^{-\alpha \frac{e^{-\frac{\theta}{x}}}{1 - e^{-\frac{\theta}{x}}}}, \quad (5)$$

where $x, \alpha, \theta > 0$. The values of the maximum likelihood estimations $\hat{\theta}$ and $\hat{\alpha}$ for the parameters θ and α , respectively, the maximum value of the log-likelihood function, the value of the Akaike information criterion (AIC) and the value of the Bayesian information criterion (BIC) are shown in Table 1.

For this data set, Table 1 shows the maximum likelihood estimation results for the inverse exponential (IE) distribution as well.

Here, we computed the maximum likelihood estimations of the parameters for the inverse epsilon distribution as follows. Let the random variable X be the

Table 1: Estimation results

Distribution	Parameters		Log-likelihood	AIC	BIC
EIE	$\hat{\theta} = 33.4469$	$\hat{\alpha} = 0.1609$	-280.4043	564.8086	568.3770
IE	$\hat{\lambda} = 76.7000$		-279.5773	561.1546	562.9389
Inverse epsilon	$\hat{\lambda} = 76.7000$	$\hat{d} = 11.9953$	-279.5773	563.1547	566.7231

survival time of patients, $X \sim \bar{\varepsilon}(\lambda, d)$, and let x_1, x_2, \dots, x_n be independent observations on X . Then, the likelihood function $L: (0, \infty)^2 \rightarrow (0, \infty)$ for the sample x_1, x_2, \dots, x_n is given by

$$L(\lambda, d; x_1, x_2, \dots, x_n) = \prod_{i=1}^n \left(\lambda \frac{d^2}{d^2 x_i^2 - 1} \left(\frac{d + x_i^{-1}}{d - x_i^{-1}} \right)^{-\lambda \frac{d}{2}} \right),$$

where $d > \frac{1}{\min_{i=1,2,\dots,n}(x_i)}$. The log-likelihood function $l = \ln \circ L$ is given by

$$l(\lambda, d; x_1, x_2, \dots, x_n) = n \ln(\lambda) + \sum_{i=1}^n \ln \left(\frac{d^2}{d^2 x_i^2 - 1} \right) - \lambda \frac{d}{2} \sum_{i=1}^n \ln \left(\frac{d + x_i^{-1}}{d - x_i^{-1}} \right),$$

where $d > \frac{1}{\min_{i=1,2,\dots,n}(x_i)}$. We used the GLOBAL method, which is a stochastic global optimization procedure introduced by Csentesi et al. [4], to find the maxima of the log-likelihood function. The estimations of λ and d , respectively, $\hat{\lambda}$ and \hat{d} , and the maximal value of the log-likelihood function are shown in Table 1.

Based on the maximum likelihood estimation results, we can summarize our findings as follows.

- (a) For the studied survival times, the inverse epsilon distribution gives better maximal log-likelihood value and better AIC and BIC values than the exponential inverse exponential distribution. At the same time, the PDF and the CDF of the inverse exponential distribution have much simpler formulas (see Eq. (1) and Eq. (2)) than those of the exponential inverse exponential distribution (see Eq. (4) and Eq. (5)).
- (b) The inverse epsilon distribution and the inverse exponential distribution result the same maximal log-likelihood value. That is, in line with the finding of Proposition 3, these two distributions coincide if $d \rightarrow \infty$. Notice that in our case, these two distributions may be viewed as being identical already for $d = 11.9953$.
- (c) The inverse epsilon distribution has two parameters (λ and d), while the inverse exponential distribution has only one parameter (λ). Therefore, as in our case these two distributions result the same maximal log-likelihood value, the AIC and BIC values for the inverse exponential distribution (561.1546 and 562.9389, respectively) are lower than those for the inverse epsilon distribution

(563.1547 and 566.7231, respectively). It should be added that by fixing the value of parameter d at a large value (e.g. $d = 100$), the inverse epsilon distribution may be treated as a one-parameter distribution, which coincides with the inverse exponential distribution.

- (d) The PDFs and the CDFs of the exponential distribution and the exponential inverse exponential distribution contain exponential terms, while the PDF and the CDF of the inverse epsilon distribution do not contain any exponential term.

Figure 4 shows the plots of the empirical CDF, EIE CDF, inverse exponential CDF and inverse epsilon CDF with the parameter values listed in Table 1.

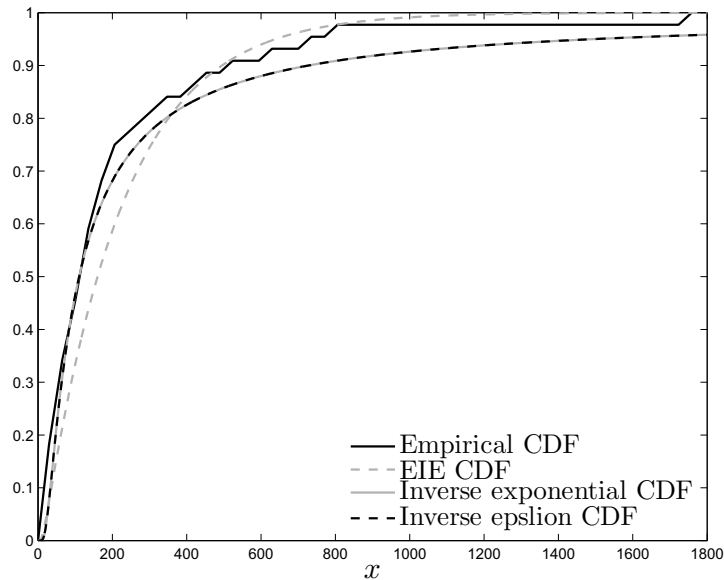


Figure 4: Empirical CDF, EIE CDF, Inverse exponential CDF and Inverse epsilon CDF

4 Conclusions

In this paper, we study the possibilities offered by a new two-parameter lower-truncated distribution constructed from the inversion of the so-called epsilon distribution. Here, diverse motivations for this new distribution are provided. We have studied in depth the shapes of the probability density and hazard rate functions, determined the quantile function and discussed the moments. The theory

is illustrated by a complete graphical analysis. Through the maximum likelihood approach, the new model is derived and an application with real data is also given.

We further plan to use the inverse epsilon distribution in an applied regression setting, and to investigate some of its natural generalizations through standard schemes (Marshall-Olkin, transmuted, type I half-logistic, etc).

Acknowledgement

We thank Dorina Keller for her assistance.

References

- [1] Aslam, Muhammad, Noor, Farzana, and Ali, Sajid. Shifted exponential distribution: Bayesian estimation, prediction and expected test time under progressive censoring. *Journal of Testing and Evaluation*, 48(2):1576–1593, 2020. DOI: 10.1520/JTE20170593.
- [2] Bai, Xuchao, Shi, Yimin, Liu, Yiming, and Zhang, Chunfang. Statistical inference for constant-stress accelerated life tests with dependent competing risks from Marshall-Olkin bivariate exponential distribution. *Quality and Reliability Engineering International*, 36(2):511–528, 2020. DOI: 10.1002/qre.2582.
- [3] Cordeiro, Gauss M, Silva, Rodrigo B, and Nascimento, Abraão DC. *Recent Advances in Lifetime and Reliability Models*. Bentham Science Publishers, 2020.
- [4] Csendes, Tibor, Pál, László, Sendin, J Oscar H, and Banga, Julio R. The GLOBAL optimization method revisited. *Optimization Letters*, 2(4):445–454, 2008. DOI: 10.1007/s11590-007-0072-3.
- [5] Dey, Sanku. Inverted exponential distribution as a life distribution model from a Bayesian viewpoint. *Data science journal*, 6:107–113, 2007. DOI: 10.2481/dsj.6.107.
- [6] Dombi, József, Jónás, Tamás, and Tóth, Zsuzsanna Eszter. The epsilon probability distribution and its application in reliability theory. *Acta Polytechnica Hungarica*, 15(1):197–216, 2018. DOI: 10.12700/APH.15.1.2018.1.12.
- [7] Efron, Bradley. Logistic regression, survival analysis, and the Kaplan-Meier curve. *Journal of the American statistical Association*, 83(402):414–425, 1988. DOI: 10.1080/01621459.1988.10478612.
- [8] Galton, Francis. *Inquiries into human faculty and its development*. Macmillan, 1883.
- [9] Gradshteyn, IS and Ryzhik, IM. *Table of Integrals, Series and Products*. Academic Press, London, 2007.

- [10] Jones, M Chris. Estimating densities, quantiles, quantile densities and density quantiles. *Annals of the Institute of Statistical Mathematics*, 44(4):721–727, 1992. DOI: 10.1007/BF00053400.
- [11] Klein, John P and Moeschberger, Melvin L. *Survival analysis: techniques for censored and truncated data*. Springer Science & Business Media, 2006.
- [12] Lin, CT, Duran, BS, and Lewis, TO. Inverted gamma as a life distribution. *Microelectronics Reliability*, 29(4):619–626, 1989. DOI: 10.1016/0026-2714(89)90352-1.
- [13] Moors, J. J. A. A quantile alternative for kurtosis. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 37(1):25–32, 1988. DOI: 10.2307/2348376.
- [14] Oguntunde, PE, Adejumo, AO, and Owoloko, Enahoro Alfred. Exponential inverse exponential (EIE) distribution with applications to lifetime data. *Asian J. Sci. Res*, 10:169–177, 2017. DOI: 10.3923/ajsr.2017.169.177.
- [15] Okorie, Idika E and Nadarajah, Saralees. On the omega probability distribution. *Quality and Reliability Engineering International*, 35(6):2045–2050, 2019. DOI: 10.1002/qre.2462.
- [16] Prajapati, Deepak, Mitra, Sharmistha, and Kundu, Debasis. A new decision theoretic sampling plan for type-I and type-I hybrid censored samples from the exponential distribution. *Sankhya B*, 81(2):251–288, 2019. DOI: 10.1007/s13571-018-0167-0.
- [17] Rastogi, Manoj K and Oguntunde, PE. Classical and bayes estimation of reliability characteristics of the Kumaraswamy-Inverse exponential distribution. *International Journal of System Assurance Engineering and Management*, 10(2):190–200, 2019. DOI: 10.1007/s13198-018-0744-7.
- [18] Singh, Sanjay Kumar, Singh, Umesh, and Kumar, Dinesh. Bayes estimators of the reliability function and parameter of inverted exponential distribution using informative and non-informative priors. *Journal of Statistical Computation and Simulation*, 83(12):2258–2269, 2013. DOI: 10.1080/00949655.2012.690156.
- [19] Yuge, Tetsushi, Maruyama, Megumi, and Yanagi, Shigeru. Reliability of a k-out-of-n system with common-cause failures using multivariate exponential distribution. *Procedia Computer Science*, 96:968–976, 2016. DOI: 10.1016/j.procs.2016.08.101.

Received 17th January 2021

An Efficient Sampling Algorithm for Difficult Tree Pairs*

Sean Cleary^a and Roland Maio^b

Abstract

It is an open question whether there exists a polynomial-time algorithm for computing the rotation distances between pairs of extended ordered binary trees. The problem of computing the rotation distance between an arbitrary pair of trees, (S, T) , can be efficiently reduced to the problem of computing the rotation distance between a difficult pair of trees (S', T') , where there is no known first step which is guaranteed to be the beginning of a minimal length path. Of interest, therefore, is how to sample such difficult pairs of trees of a fixed size. We show that it is possible to do so efficiently, and present such an algorithm that runs in time $O(n^4)$.

Keywords: rotation distances, associahedra, rooted binary trees, sampling

1 Introduction

Trees are a fundamental data structure with wide applications ranging from efficient search (such as binary search trees) to modeling biological processes (such as phylogenetic trees). Given pairs of trees, there are numerous ways of calculating metrics of interest between trees. These metrics measure of the amount of commonality and difference, which depend on the class of trees considered and what features of the trees are regarded as important to have in common.

Trees arise in data storage and searching as efficient structures. When there is a natural order on leaves, we have binary search trees which underlie many storage and searching systems. See Knuth [10] for background and important foundational notions and algorithms. Binary search trees and their generalizations underlie almost all modern file-storage and data-storage systems. To ensure good average-time search performance of $\log(n)$, it is necessary to have reasonably balanced trees and rotations are a quick, local change which can be used to keep trees close to

*Partial funding provided by NSF #1417820. This work was partially supported by a grant from the Simons Foundation (#234548 to Sean Cleary).

^aDepartment of Mathematics, The City College of New York and the CUNY Graduate Center, USA, E-mail: scleary@ccny.cuny.edu, ORCID: 0000-0002-3123-8658

^bDepartment of Computer Science, Columbia University, New York, NY, USA, E-mail: rolandmaio38@gmail.com, ORCID: 0000-0002-7317-770X

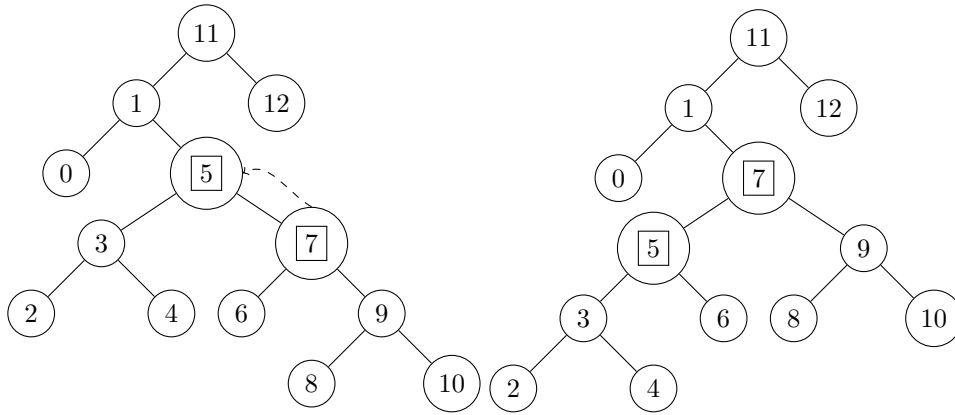


Figure 1: An example of a left rotation in an ordered tree. We rotate leftward at node 5, with the rotation promoting node 7 to the position of its former parent. Node 5 is demoted to become a left child of node 7 and node 7 is promoted to be the right child of node 1. Node 6 changes from being the left child of 7 to the right child of 5.

balanced during sequences of insertions and deletions. Furthermore, rooted binary trees are in direct correspondence with triangulations of polygons with a marked edge, and as described by Hanke, Ottmann, and Schuierer [9], such tree metrics apply to edge-flipping conversions between triangulations of planar regions.

One widely-considered tree distance metric on trees with a natural left-to-right order on leaves is that of the rotation distance between a pair of extended ordered binary trees. A rotation at a node is a simple local transformation which does not affect the order, with an example shown in Figure 1. Rotation distance between trees is the length of the shortest possible sequence of rotations to accomplish the transformation between trees. There are no known polynomial-time algorithms for computing rotation distance. Culik and Wood [7] originally described rotation distance, and ground-breaking work of Sleator, Tarjan and Thurston [13] used the correspondence between trees with n internal nodes and triangulations of the marked regular $n + 2$ -gon to show that if there is a common edge between the two triangulations then any shortest path does not flip this edge. Such a common edge thus breaks the rotation distance problem into two smaller sub-problems. Furthermore, they showed that if it is possible to flip an edge of either polygon to obtain a common edge, then there is a shortest path which begins by doing so. We call edges which are not common but which can be flipped to become a common edge *one-off* edges, as they are one move away from being common edges themselves. Cleary and St. John [6] used these reduction rules to show that the rotation distance problem is fixed parameter tractable.

We call a pair of trees with no common edges and no edges which can be immediately flipped to create a common edge a *difficult tree pair*. The above

reductions transform the problem of computing the rotation distance on a pair of trees drawn from all possible pairs to a pair of trees drawn from the set of all such difficult tree pairs. A common edge, arising either immediately or from performing a single flip to change a one-off edge to a common edge, then naturally splits the tree pair into a pair of smaller tree pairs, as explained in Sleator, Tarjan, and Thurston [13]. The kernel of the difficult of the rotation distance problem at this point is to find distances between difficult tree pairs.

To understand how effective different approximation and partial algorithms are at evaluating and estimating rotation distance, it would be useful to sample difficult tree pairs. It is possible to find examples of difficult tree pairs by picking a tree pair of large size at random, and then performing all possible reductions and one-off moves, splitting the problem into a collection of smaller subproblems, until either the trees are identical (extremely unlikely) or until a collection of difficult tree pairs is obtained. But such a procedure is not only time-consuming, it is not possible to tell in advance how many reductions there will be and what the resulting sizes of the smaller remaining difficult piece pairs will be. Thus there is no control on the resulting size of the difficult tree pairs produced. In general (see Cleary, Reznitzer and Wong [2]) there are a sizable number of common edges and one-off edges, resulting on average about at least a 10% reduction in the size of a randomly selected tree pair to a largest difficult remaining tree pair. It is not difficult to construct specific examples of specified size of difficult tree pairs- examples of Dehornoy [8], Pournin [11], and Cleary and Maio [3] are families of difficult pairs but in each case of a restricted type. In many of these very specific cases, analysis to that family of instances can give coincident upper and lower bounds on rotation distance, giving an exact calculation. But these families are very sparse in the set of all difficult tree pairs. The set of all difficult tree pairs appears to grow exponentially with size, but at a slower exponential growth rate than the set of all tree pairs, per work of Cleary and Maio [4] suggesting that the fraction of all tree pairs decreases exponentially at a rate of about 0.77^n , with already ratio of less than 1 in a billion tree pairs of size 70 being difficult and the fraction dropping with further increases of size.

Difficult tree pairs lie at the kernel of a number of questions of interest. Because the rotation distance problem frequently splits into smaller subproblems, the essential difficulties are contained in the set of difficult tree pairs. Difficult tree pairs can be used to test estimation algorithms for rotation distance, to find estimates for typical rotation distance between tree pairs selected at random, and to look for difficult pathological behavior for rotation distance paths.

This motivates studying difficult tree pairs in their own right. We describe below an efficient algorithm for sampling difficult tree pairs of a specified size. This sampling is not uniform across all difficult tree pairs of a prescribed size but does have wide coverage of such pairs.

The algorithm we describe can be seen as a variation on Remy's algorithm [12] for efficiently generating rooted ordered trees uniformly at random, but instead of working on growing the size of a single tree, we grow a pair of trees while applying a filtering criterion. Unlike Remy's algorithm, the difficult pairs are not sampled uniformly at random but having an efficient (polynomial-time) means of generating

pairs is useful for understanding rotation distance problem instances better and for testing the performance of new algorithms. Computational experiments show that the distribution of selected tree pairs are not uniformly random but there does seem to be wide dispersion, with relatively broad coverage of difficult tree pairs.

2 Background

An *extended ordered binary tree* is a rooted binary tree where every node has exactly 0 or 2 children and whose leaves are labelled starting with 0 in their order defined by a pre-order traversal from the root. The label of a leaf node ℓ is denoted $\text{label}(\ell)$. The size of an extended ordered binary tree T , denoted $|T|$, is the number of internal nodes T contains. The set of all nodes in T is denoted $\text{nodes}(T)$. In the following *tree* will refer to an extended ordered binary tree and S and T will be trees of the same size.

A *rotation at a node ν* in a tree is a local operation which promotes an internal node ν to the position of its parent μ , demotes μ to one of ν 's children, and makes one of ν 's children a child of μ , illustrated in Figure 1 where the rotation is leftward at node 7 going from the left tree to the right tree. The inverse operation of going from the right tree to the left tree is a rightward rotation at node 5. We will denote the partial function that returns the parent of a node by $\pi : \text{nodes}(T) \mapsto \text{nodes}(T)$, with $\pi(\text{root}(T))$ is undefined. We adopt the convention that the statement “rotate at a node ν ”, denoted $\text{rotate}(\nu)$, means to perform that rotation which promotes ν to the position of its parent. Whether that is a right or left rotation depends upon whether ν is a left or right child of its parent. For a tree of size n there are $n - 1$ possible rotations, one for each internal node except the root.

Given a pair of trees (S, T) of the same size, it is possible to transform one into the other by some sequences of rotations. The minimum length of any such sequence defines the *rotation distance* between S and T , which we denote $d(S, T)$.

The interval of a node ν , $\text{interval}(\nu)$, is the pair (α, β) where α is the label of the least-labelled leaf in the tree rooted at ν and β is the label of the greatest-labelled leaf in the tree rooted at ν . The label α is called the *lower bound of the interval of ν* and is denoted $\lfloor \text{interval}(\nu) \rfloor$. Similarly, the label β is called the *upper bound of the interval of ν* and is denoted $\lceil \text{interval}(\nu) \rceil$. If ν is a leaf, then its lower bound is the same as its upper bound and is defined to be its label. If ν is an internal node, then its lower bound is the lower bound of its left child, and its upper bound is the upper bound of its right child; formally

$$\text{interval}(\nu) = \begin{cases} (\text{label}(\nu), \text{label}(\nu)) & \text{if } \nu \text{ is a leaf} \\ (\lfloor \text{interval}(\text{left}(\nu)) \rfloor, \lceil \text{interval}(\text{right}(\nu)) \rceil) & \text{otherwise} \end{cases}$$

The intervals of a tree T , denoted $\text{intervals}(T)$, is the set of all intervals of the internal nodes of T .

The labels α and β are related to each other by the size of the subtree rooted at ν in the following way:

Proposition 1. *Let ν be an internal node of T , and N the subtree rooted at ν , and $(\alpha, \beta) = \text{interval}(\nu)$, then $\beta = \alpha + |N|$.*

Proof. Recall that N has $|N| + 1$ leaves. It is a property of pre-order traversal that once the traversal visits a node it will visit the entire subtree rooted at that node before it visits any other part of the tree. Consequently, when the pre-order traversal reaches ν , the next $|N| + 1$ leaf nodes that will be visited will be the leaf nodes of N . Thus, the greatest label any leaf in N can have is $\alpha + |N|$ and this must be attained by the last leaf that is visited in N . \square

In addition to changing one tree into another, a rotation in a tree T at a node ν has the effect of replacing one of the intervals of the tree by a new one. This new interval is uniquely determined by T and ν and is denoted $1\text{-interval}(\nu)$. The $1\text{-interval}(\nu)$ can be defined in terms of the intervals of ν , the parent of ν , and the children of ν . If ν is the left child of its parent, then the lower bound of $1\text{-interval}(\nu)$ is the lower bound of the right child of ν and the upper bound of $1\text{-interval}(\nu)$ is the upper bound of the parent of ν . If ν is the right child of its parent, then the lower bound of $1\text{-interval}(\nu)$ is the lower bound of its parent, and the upper bound of $1\text{-interval}(\nu)$ is the upper bound of the left child of ν . Formally

$$1\text{-interval}(\nu) = \begin{cases} ([\text{interval}(t)], \lceil \text{interval}(\pi(\nu)) \rceil) & \text{if } \nu = \text{left}(\pi(\nu)) \\ ([\text{interval}(\pi(\nu))], \lceil \text{interval}(s) \rceil) & \text{otherwise} \end{cases}$$

where $s = \text{left}(\nu)$ and $t = \text{right}(\nu)$.

The $1\text{-intervals}(T)$ is the set of all $n - 1$ intervals that can be obtained by rotating some node in T .

Trees correspond naturally to the marked triangulations of a polygon, and we denote the corresponding triangulation by $\Delta(T)$. The edges of $\Delta(T)$ correspond to the intervals(T).

While the reduction rules were first developed from the perspective of triangulations of the polygon, they may be formulated from the tree perspective in terms of intervals and rotations. A common edge between triangulations corresponds to a common interval occurring in the intervals of both trees. A one-off edge between triangulations corresponds to a common interval that can be obtained by rotating at one of the nodes in S or T .

The binary word of T , $\text{word}(T)$, is obtained by beginning with the empty string, traversing T in pre-order and appending at each node a ‘1’ if the node is an internal node and a ‘0’ otherwise. Thus the symbol at the i th index in $\text{word}(T)$ is determined by the i th node visited in T by a pre-order traversal. This determines a mapping from symbols in $\text{word}(T)$ to nodes(T).

Definition 1. *Let T be an extended ordered binary tree, and let ν be the i -th node visited in a pre-order traversal of T . The symbol of ν in $\text{word}(T)$, denoted $\text{sym}_T(\nu)$, is defined to be the symbol of $\text{word}(T)$ at index i .*

The following property of $\text{word}(T)$ gives one method for computing the intervals of T .

Proposition 2. *Let ℓ be a leaf node of T , then the label of ℓ is given by the number of ‘0’s that precede $\text{sym}_T(\ell)$.*

Proof. Suppose the label of ℓ is α . By the definition of label, ℓ is the $(\alpha + 1)$ st leaf node visited in the preorder traversal of T . In computing $\text{word}(T)$, therefore, exactly α ‘0’s must have been appended before $\text{sym}_T(\ell)$ is appended. \square

Proposition 3. *Let T be an extended ordered binary tree, ν an internal node of T , N the subtree of T rooted at ν , and $(\alpha, \beta) = \text{interval}(\nu)$. Then α is given by the number of 0’s that precede $\text{sym}_T(\nu)$, and $\beta = \alpha + |N|$.*

Proof. To prove α is given by the number of 0’s that precede $\text{sym}_T(\nu)$ it suffices, by Proposition 2, to show that the symbol of the leaf node, ℓ , with label α is the first 0 that proceeds $\text{sym}_T(\nu)$. Suppose this is not the case, then there is at least one 0 that proceeds $\text{sym}_T(\nu)$ and precedes $\text{sym}_T(\ell)$. Then there is some leaf node k in the subtree rooted at ν that is visited after ν and before ℓ . So the label of k is at most $\alpha - 1$, but this contradicts the assumption that ℓ is the least labelled leaf. Finally, from Proposition 1 it follows that $\beta = \alpha + |N|$. \square

We let \circ denote string concatenation and we let ν be a node of a tree. We define the functions $\text{left}(\nu)$ and $\text{right}(\nu)$ to return the left or right child of ν respectively. A recursive definition for $\text{word}(\nu)$ can then be given as follows

$$\text{word}(\nu) = \begin{cases} 1 \circ \text{word}(\text{left}(\nu)) \circ \text{word}(\text{right}(\nu)) & \text{if } \nu \text{ is an internal node} \\ 0 & \text{otherwise} \end{cases}$$

With this definition $\text{word}(T) = \text{word}(r)$ where r is the root of T .

Remy’s algorithm [12] is a method for sampling trees of a fixed size uniformly at random by growing a tree larger at each stage ensuring that each possible tree of that size is equally likely to be generated. The algorithm begins with a tree of size 1 and iteratively grows the tree until a tree of the desired size is obtained. On each iteration, one of the internal or external nodes, say ν , of the current tree, say T , is selected uniformly at random. Then a new node, μ , is created. The new node μ takes the place of ν in the tree, and ν is set as the left or right child of μ with equal probability. We say that the resulting tree is obtained from T by growing left (or right) at ν .

If a tree S may be grown in some way by an iteration of Remy’s algorithm to obtain a tree T , then we call T a growth neighbor of S and denote the set of all growth neighbors of S by $\text{growthNeighbors}(S)$.

On an iteration of Remy’s algorithm, if an external node is chosen to be grown, then growing left or right will result in the same tree. Thus, an upper bound on the number of growth neighbors a tree of size n may have is $3n + 1$.

3 Difficult Pair Sampling Algorithm

The Difficult Pair Sampling algorithm, DPS, begins by randomly choosing one of the 4 difficult pairs of trees of size 4. We call these difficult pairs *primitive* because

there are no difficult pairs of trees that are smaller. The algorithm then iteratively grows the pair of trees in size by 1 until a pair of the desired size is obtained. On each iteration, for the current pair of trees S and T , DPS finds all difficult pairs of trees (U, V) such that U is a growth neighbor of S and V is a growth neighbor of T and randomly selects one of these pairs to be the next S and T .

```

DPS( $n$ )
1  $S, T = \text{randomPrimitiveDifficultPair}()$ 
2 for  $i = 5$  to  $n$ 
3      $\text{choices} = \emptyset$ 
4     for  $U$  in  $\text{growthNeighbors}(S)$ 
5         for  $V$  in  $\text{growthNeighbors}(T)$ 
6             if  $\text{isDifficultPair}(U, V)$ 
7                  $\text{choices.add}((U, V))$ 
8      $S, T = \text{choices.randomElement}()$ 
9 return  $S, T$ 
    
```

What is not obvious about DPS is that for an arbitrary difficult pair (S, T) , it is always possible to grow S and T into a difficult pair (U, V) . We will show that this is the case by examining a particular growth neighbor. Generally there are many additional growth neighbors but the existence of a single one suffices for proving the correctness of the algorithm.

Definition 2. Let T be an extended ordered binary tree of size n , and ω be the internal node of T whose right child is the leaf with label n . The extended ordered binary tree of size $n + 1$, obtained by growing T at ω left, will be denoted $\sigma(T)$.

We will show that given a difficult pair (S, T) , the pair of trees $(\sigma(S), \sigma(T))$ is also a difficult pair. The proof that $(\sigma(S), \sigma(T))$ is a hard pair will rest on the relation between $\text{intervals}(T)$ to $\text{intervals}(\sigma(T))$ and $1\text{-intervals}(T)$ to $1\text{-intervals}(\sigma(T))$.

Relating $\text{intervals}(T)$ to $\text{intervals}(\sigma(T))$ and $1\text{-intervals}(T)$ to $1\text{-intervals}(\sigma(T))$ will require relating $\text{word}(T)$ to $\text{word}(\sigma(T))$ which we will do next.

Lemma 1. Let T be an extended ordered binary tree of size n , then $\text{word}(T) = \Lambda\Omega$ and $\text{word}(\sigma(T)) = \Lambda 1\Omega 0$.

Proof. Let ω be the parent of the leaf node with label n in T , this implies that ω , and all of its ancestors are either the root or the right child of their parent. From the definition of word it follows that $\text{word}(T)$ is of the form $\Lambda\Omega$ where $\Omega = \text{word}(\omega)$.

When T is grown left at ω , the new node, ϕ , will take ω as its left child and become the right child of ω 's former parent. Consequently, $\text{word}(\sigma(T))$ will be $\Lambda\Phi$ where $\Phi = \text{word}(\phi)$.

$$\begin{aligned}
 \text{word}(\phi) &= 1 \circ \text{word}(\text{left}(\phi)) \circ \text{word}(\text{right}(\phi)) \\
 &= 1 \circ \text{word}(\omega) \circ 0 \\
 \Phi &= 1\Omega 0
 \end{aligned}$$

□

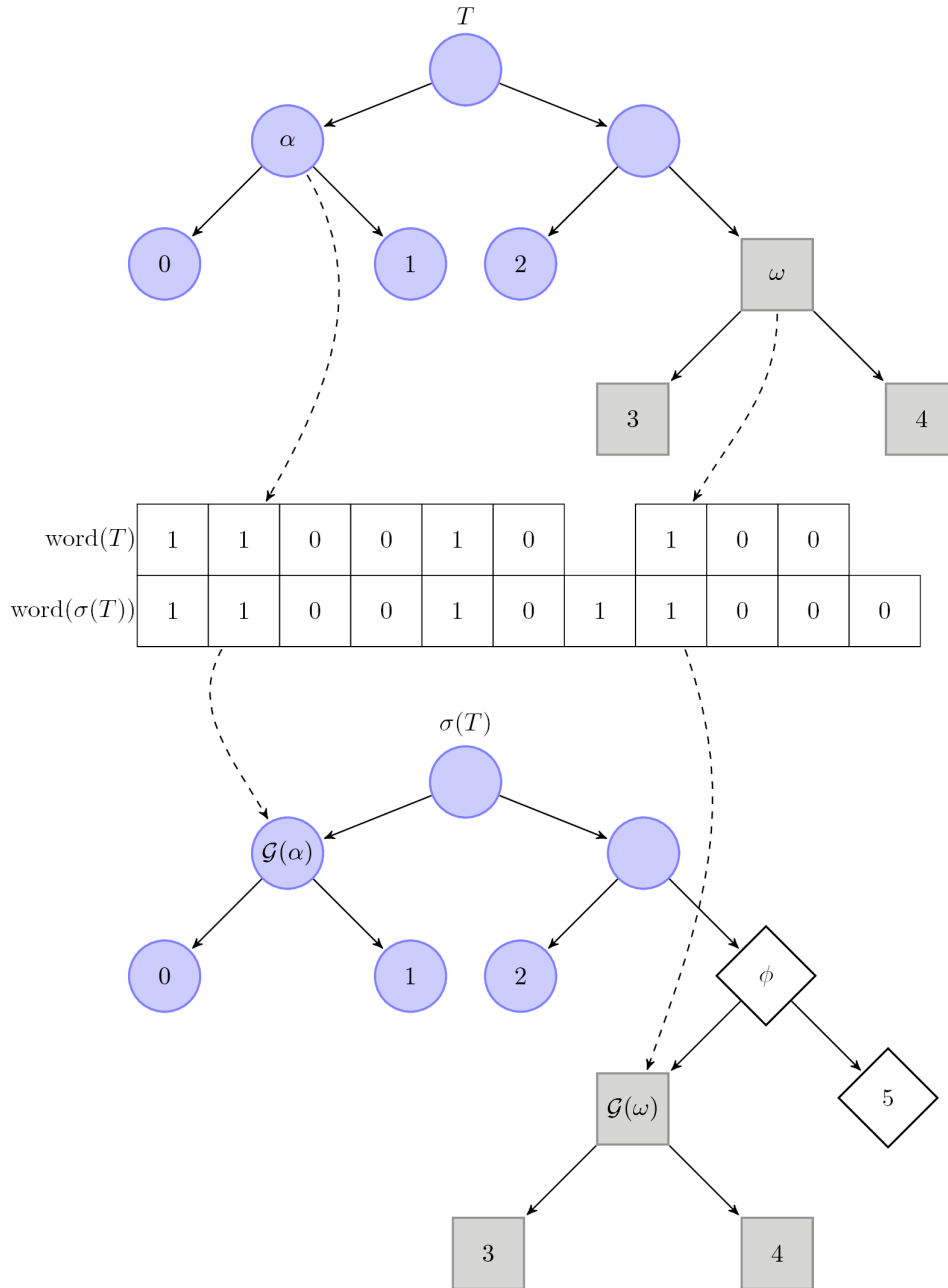


Figure 2: Growing a node and preserving the difficulty of the pair.

There is a natural way in which the nodes in T correspond to the nodes in $\sigma(T)$. The intuition for this is shown in Figure 2. This correspondence can be formalized in terms of the relation between $\text{word}(T)$ and $\text{word}(\sigma(T))$.

Definition 3. Let T be an extended ordered binary tree. The natural growth injection of the nodes of T to the nodes of $\sigma(T)$, $\mathcal{G} : \text{nodes}(T) \mapsto \text{nodes}(\sigma(T))$ is defined as

$$\mathcal{G}(\nu) = \text{sym}_{\sigma(T)}^{-1}(\text{word}(\sigma(T))(i + \mathbf{1}\{|\Lambda| < i\}))$$

where i is the index of $\text{sym}_T(\nu)$ and $\mathbf{1}\{\dots\}$ is the indicator function.

There are several properties of \mathcal{G} which will be critical to proving our claim that DPS can always grow a difficult pair (S, T) into another difficult pair (U, V) . The first that we will examine relates the interval of a node, ν , in T to the interval of $\mathcal{G}(\nu)$ in $\sigma(T)$.

Lemma 2. Let T be an extended ordered binary tree of size n , ω be the node of T whose right child is the leaf with label n , ν be any node of T that is not ω and $(\alpha, \beta) = \text{interval}(\nu)$. Then $\text{interval}(\mathcal{G}(\nu)) = (\alpha, \beta + \mathbf{1}\{\beta = n\})$.

Proof. Let $(\gamma, n) = \text{interval}(\omega)$. By Lemma 1, we have $\text{word}(T) = \Lambda\Omega$ and $\text{word}(\sigma(T)) = \Lambda 1\Omega 0$. Now we partition $\text{nodes}(T) - \{\omega\}$ into three sets:

1. $\{\nu : (\alpha, \beta) = \text{interval}(\nu), \alpha < \gamma, \beta < n\}$ all nodes that have an interval with a lower bound less than the lower bound of ω and with an upper bound that is less than n .
2. $\{\nu : (\alpha, \beta) = \text{interval}(\nu), \alpha < \gamma, \beta = n\}$ all nodes that have an interval with a lower bound less than the lower bound of ω and with an upper bound equal to n .
3. $\{\nu : (\alpha, \beta) = \text{interval}(\nu), \gamma \leq \alpha\}$ all nodes that have an interval with a lower bound greater than or equal to the lower bound of ω .

We consider the first case. Let N be the subtree rooted at ν , a be the leaf with label α , b be the leaf with label β and g be the leaf with label γ . Since $\alpha < \gamma$, the symbol of a in $\text{word}(T)$ must precede the symbol of g in $\text{word}(T)$ and therefore $\text{sym}_T(a) \in \Lambda$ and $\text{sym}_T(\nu) \in \Lambda$. By the definition of \mathcal{G} it follows that the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu))$ is the same as the index of $\text{sym}_T(\nu)$ and so $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu)) \in \Lambda$. Therefore, by Proposition 2 the lower bound of $\mathcal{G}(\nu)$ must be α . Since $\beta < n$ it follows $\beta < \gamma$, otherwise g is in N , but this would imply that ω is also in N and so $\beta = n$, which is impossible. Consequently the symbol of b in $\text{word}(T)$ must precede the symbol of g in $\text{word}(T)$ and therefore $\text{sym}_T(b) \in \Lambda$ and $\text{sym}_{\sigma(T)}(\mathcal{G}(b)) \in \Lambda$. This implies that $\text{word}(\nu) = \text{word}(\mathcal{G}(\nu))$ and so the size of the subtree rooted at $\mathcal{G}(\nu)$ is the same as the size of N . Applying Theorem 3 it follows that the upper bound of $\mathcal{G}(\nu)$ is β . Therefore $\text{interval}(\mathcal{G}(\nu)) = (\alpha, \beta) = (\alpha, \beta + \mathbf{1}\{\beta = n\})$.

Now we consider the second case. Let N be the subtree rooted at ν , a be the leaf with label α , b be the leaf with label β and g be the leaf with label γ . Since $\alpha < \gamma$, the symbol of a in $\text{word}(T)$ must precede the symbol of g in $\text{word}(T)$ and so

$\text{sym}_T(a) \in \Lambda$ and $\text{sym}_T(\nu) \in \Lambda$. By the definition of \mathcal{G} , the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu))$ is the same as the index of $\text{sym}_T(\nu)$ and by Proposition 2 the lower bound of $\mathcal{G}(\nu)$ is α . Since $\beta = n$, ω must be contained in the subtree rooted at ν . Therefore, when T is grown left at ω , the subtree rooted at $\mathcal{G}(\nu)$ will be larger in size by one than N . Applying Theorem 3 it follows that the upper bound of $\mathcal{G}(\nu)$ is $\beta + 1$. And so $\text{interval}(\mathcal{G}(\nu)) = (\alpha, \beta + 1) = (\alpha, \beta + \mathbf{1}\{\beta = n\})$.

Finally, we consider the third case. Let N be the subtree rooted at ν . Since $\gamma \leq \alpha$, ν is a descendant of ω . Therefore $\text{word}(\nu)$ is a proper substring of Ω . We observe that for $|\Lambda| < i \leq |\text{word}(T)|$, we have $\text{word}(T)(i) = \text{word}(\sigma(T))(i + 1)$, combined with the definition of \mathcal{G} it follows that $\text{word}(\nu) = \text{word}(\mathcal{G}(\nu))$. So the size of the subtree rooted at $\mathcal{G}(\nu)$ is the same as the size of N . Since the number of '0's which precede $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu))$ is the same as the number of '0's that precede $\text{sym}_T(\nu)$ it follows that the lower bound of $\mathcal{G}(\nu)$ is α . Therefore $\text{interval}(\mathcal{G}(\nu)) = (\alpha, \beta) = (\alpha, \beta + \mathbf{1}\{\beta = n\})$. □

The growth injection \mathcal{G} also captures several of the node-to-node relationships of T which are preserved in $\sigma(T)$: \mathcal{G} preserves the relation between parents and their left children, \mathcal{G} preserves the relation between parents and their right children, except for the parent of the node whose right child is the leaf labelled n , and taken together, a consequence of the preceding two properties is that \mathcal{G} preserves the relation between parents and children except for the node whose right child is the leaf labelled n . We will next state and prove these relationships formally because we will exploit them in proving the relationship between 1-intervals(T) and 1-intervals($\sigma(T)$).

Lemma 3. *Let T be an extended ordered binary tree of size n and ω be the internal node of T whose right child is the leaf with label n . For any internal node $\nu \in T$, the image of the left child of ν under \mathcal{G} is the left child of the image of ν under \mathcal{G} , that is, $\mathcal{G}(\text{left}(\nu)) = \text{left}(\mathcal{G}(\nu))$.*

Proof. It will suffice to show that the index of the symbol of $\mathcal{G}(\text{left}(\nu))$ in the word of $\sigma(T)$ is the same as the index of the symbol of $\text{left}(\mathcal{G}(\nu))$ in the word of $\sigma(T)$. Let i be the index of $\text{sym}_T(\nu)$ and consider two cases as to whether $|\Lambda| < i$ or not.

Case I: $|\Lambda| < i$: the index of $\text{sym}_T(\text{left}(\nu)) = i + 1$ and so the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\text{left}(\nu))) = i + 2$. By definition, the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu)) = i + 1$ giving index $\text{sym}_{\sigma(T)}(\text{left}(\mathcal{G}(\nu))) = i + 2$.

Case II: $|\Lambda| \geq i$; we must verify that $i + 1 \leq |\Lambda|$. Since ν is an internal node, its symbol must be a '1' and since the suffix of Λ is the word of the left child of the parent of ω , the last symbol in Λ must be a '0' and so $i \leq |\Lambda| - 1$ which implies $i + 1 \leq |\Lambda|$. Now the index of $\text{sym}_T(\text{left}(\nu)) = i + 1$. Since $i + 1 < |\Lambda|$ the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\text{left}(\nu))) = i + 1$. By definition, the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu)) = i$ and so the index of $\text{sym}_{\sigma(T)}(\text{left}(\mathcal{G}(\nu))) = i + 1$. □

Lemma 4. *Let T be an extended ordered binary tree of size n and ω be the internal node of T whose right child is the leaf with label n . For any internal node $\nu \in T$,*

except for $\pi(\omega)$, the image of the right child of ν under \mathcal{G} is the right child of the image of ν under \mathcal{G} , that is $\mathcal{G}(\text{right}(\nu)) = \text{right}(\mathcal{G}(\nu))$.

Proof. It will suffice to show that the index of the symbol of $\mathcal{G}(\text{right}(\nu))$ in the word of $\sigma(T)$ is the same as the index of the symbol of $\text{right}(\mathcal{G}(\nu))$ in the word of $\sigma(T)$. Let i be the index of $\text{sym}_T(\nu)$ and consider two cases as to whether $|\Lambda| < i$ or not.

Case I: $|\Lambda| < i$: the index of $\text{sym}_T(\text{right}(\nu)) = i + |\text{word}(\text{left}(\nu))| + 1$ and so the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\text{right}(\nu))) = i + |\text{word}(\text{left}(\nu))| + 2$. By definition, the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu)) = i + 1$ and so the index of $\text{sym}_{\sigma(T)}(\text{right}(\mathcal{G}(\nu))) = i + |\text{word}(\text{left}(\mathcal{G}(\nu)))| + 2$. Observe that the upper bound of the interval of $\text{left}(\nu)$ must be less than n since it is the left child of its parent. Applying Lemmas 2 and 3 it follows that $\text{interval}(\text{left}(\nu)) = \text{interval}(\mathcal{G}(\text{left}(\nu))) = \text{interval}(\text{left}(\mathcal{G}(\nu)))$. Therefore the size of the subtree rooted at $\text{left}(\nu)$ is the same as the size of the subtree rooted at $\text{left}(\mathcal{G}(\nu))$ and so $|\text{word}(\text{left}(\nu))| = |\text{word}(\text{left}(\mathcal{G}(\nu)))|$.

Case II: $|\Lambda| \geq i$: we must verify that $i + |\text{word}(\text{left}(\nu))| + 1 \leq |\Lambda|$. But this is clearly true since the only case in which $|\Lambda| < i + |\text{word}(\text{left}(\nu))| + 1$ is when $\nu = \pi(\omega)$, but we have excluded $\pi(\omega)$ from consideration. Observe that the index of $\text{sym}_T(\text{right}(\nu)) = i + |\text{word}(\text{left}(\nu))| + 1$ and so the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\text{right}(\nu))) = i + |\text{word}(\text{left}(\nu))| + 1$. By definition, the index of $\text{sym}_{\sigma(T)}(\mathcal{G}(\nu)) = i$ and so the index of $\text{sym}_{\sigma(T)}(\text{right}(\mathcal{G}(\nu))) = i + |\text{word}(\text{left}(\mathcal{G}(\nu)))| + 1$. Since $\text{left}(\nu)$ is a left child the upper bound of its interval must be less than n . Applying lemmas 2 and 3 it follows that $\text{interval}(\text{left}(\nu)) = \text{interval}(\mathcal{G}(\text{left}(\nu))) = \text{interval}(\text{left}(\mathcal{G}(\nu)))$. This implies that the size of the subtree rooted at $\text{left}(\nu)$ is the same as the size of the subtree rooted at $\text{left}(\mathcal{G}(\nu))$ and so $|\text{word}(\text{left}(\nu))| = |\text{word}(\text{left}(\mathcal{G}(\nu)))|$. \square

Together Lemma 3 and Lemma 4 show the preserved parent structure.

Corollary 1. *Let T be an extended ordered binary tree of size n and ω be the internal node of T whose right child is the leaf with label n . For any node $\nu \in \text{nodes}(T) - \{\omega\}$, the image of the parent of ν under \mathcal{G} is the parent of the image of ν under \mathcal{G} , that is, $\mathcal{G}(\pi(\nu)) = \pi(\mathcal{G}(\nu))$.*

The next lemma will relate $\text{intervals}(T)$ to $\text{intervals}(\sigma(T))$.

Lemma 5. *Let T be an extended ordered binary tree of size n and ω be the internal node of T whose right child is the leaf with label n . Then the intervals of $\sigma(T)$ are related to the intervals of T by*

$$\text{intervals}(\sigma(T)) = \{(\alpha, \beta + \mathbf{1}\{\beta = n\}) : (\alpha, \beta) \in \text{intervals}(T)\} \cup \{\text{interval}(\omega)\}. \quad (1)$$

Proof. Let $(\gamma, n) = \text{interval}(\omega)$. By Lemma 2 we have the intervals for the internal nodes of $\sigma(T)$ that are the image under \mathcal{G} of some node in T , except for ω . This gives us $n - 1$ intervals of $\sigma(T)$ and we have only to consider the interval of $\mathcal{G}(\omega)$ and the interval of $\pi(\mathcal{G}(\omega))$. By Lemma 1 the number of ‘0’s which precede $\text{sym}_{\sigma(T)}(\mathcal{G}(\omega))$ is the same as the number of ‘0’s which precede $\text{sym}_{\sigma(T)}(\pi(\mathcal{G}(\omega)))$ which is the same as the number of ‘0’s which precede $\text{sym}_T(\omega)$ and so the intervals of $\mathcal{G}(\omega)$ and

$\pi(\mathcal{G}(\omega))$ have the same lower bound, namely γ , which is the lower bound of ω . By construction, the subtree rooted at $\mathcal{G}(\omega)$ has the same size as the subtree rooted at ω . Applying Proposition 1 it follows that $\text{interval}(\mathcal{G}(\omega)) = \text{interval}(\omega)$. Also by construction, the subtree rooted at $\pi(\mathcal{G}(\omega))$ is greater in size by 1 than the subtree rooted at ω . Applying Proposition 1 again yields $\text{interval}(\pi(\mathcal{G}(\omega))) = (\gamma, n + 1) = (\alpha, \beta + \mathbf{1}\{\beta = n\})$. \square

With the relationship between $\text{intervals}(\sigma(T))$ and $\text{intervals}(T)$ proven, we can state and prove the relationship between $1\text{-intervals}(\sigma(T))$ and $1\text{-intervals}(T)$. Our proof strategy will be to determine for each internal node of T , ν , the local structure that determines the $1\text{-interval}(\nu)$ in T . Then we will determine how that local structure maps to the local structure of $\mathcal{G}(\nu)$ in $\sigma(T)$ and use this to compute $1\text{-interval}(\mathcal{G}(\nu))$.

Lemma 6. *Let T be an extended ordered binary tree of size n , ω the internal node of T whose right child is the leaf with label n , and ϕ the internal node of $\sigma(T)$ that is the parent of $\mathcal{G}(\omega)$. Then the 1-intervals of $\sigma(T)$ are related to the 1-intervals of T by*

$$1\text{-intervals}(\sigma(T)) = \{(\alpha, \beta + \mathbf{1}\{\beta = n\}) : (\alpha, \beta) \in \Theta\} \cup \{1\text{-interval}(\text{left}(\omega)), (n, n + 1), 1\text{-interval}(\phi)\}$$

where $\Theta = 1\text{-intervals}(T) - \{1\text{-interval}(\omega), 1\text{-interval}(\text{left}(\omega))\}$.

Proof. Let $(\gamma, n) = \text{interval}(\omega)$ and apply lemma 1 to obtain $\text{word}(T) = \Lambda\Omega$ and $\text{word}(\sigma(T)) = \Lambda 1\Omega 0$. Now we partition nodes(T) into 6 sets:

1. $\{\nu : (\delta, n) = \text{interval}(\nu), \nu \notin \{\omega, \text{root}(T)\}\}$ every node, excluding the root and ω , whose interval has an upperbound of n .
2. $\{\nu : (\delta, \beta) = \text{interval}(\pi(\nu)), \beta < n, \nu = \text{left}(\pi(\nu))\}$ every node whose parent's interval has an upperbound less than n and that is a left child of its parent.
3. $\{\nu : (\alpha, \delta) = \text{interval}(\pi(\nu)), \delta < n, \nu = \text{right}(\pi(\nu))\}$ every node whose parent's interval has an upperbound less than n and that is a right child of its parent.
4. $\{\nu : (\delta, n) = \text{interval}(\pi(\nu)), (\kappa, n) \neq \text{interval}(\nu), \nu \neq \text{left}(\omega)\}$ every node, ν , excluding the left child of ω , such that the upper bound of the interval of ν is not n and the upper bound of the interval of ν 's parent is n .
5. $\{\omega\}$ the singleton set containing ω .
6. $\{\text{left}(\omega)\}$ the singleton set containing the left child of ω .

We proceed analyzing these six sets:

Case 1: Because we have excluded the root, the parent of ν is a node in T and its right child must be ν . Since the $1\text{-interval}(\nu) = (\alpha, \beta)$ is obtained by taking

the lower bound of ν 's parent and the upper bound of the left child of ν it follows that $\text{interval}(\pi(\nu)) = (\alpha, n)$ and $\text{interval}(\text{left}(\nu)) = (\delta, \beta)$. Since we have excluded ω , α , β and δ must be less than γ . Applying Lemma 2, Lemma 3 and Corollary 1 we have $\text{interval}(\mathcal{G}(\nu)) = (\delta, n + 1)$, $\text{interval}(\mathcal{G}(\pi(\nu))) = \text{interval}(\pi(\mathcal{G}(\nu))) = (\alpha, n + 1)$ and $\text{interval}(\mathcal{G}(\text{left}(\nu))) = \text{interval}(\text{left}(\mathcal{G}(\nu))) = (\delta, \beta)$. It follows that $1\text{-interval}(\mathcal{G}(\nu)) = (\alpha, \beta) = (\alpha, \beta + \mathbf{1}\{\beta = n\}) = 1\text{-interval}(\nu)$.

Case 2: $\text{interval}(\nu) = (\delta, \epsilon)$ and $\text{interval}(\text{right}(\nu)) = (\alpha, \epsilon)$ for some ϵ and α . Since $\beta < n$, applying Lemma 2, Lemma 4 and Corollary 1 yields $\text{interval}(\mathcal{G}(\nu)) = \text{interval}(\nu)$, as well as $\text{interval}(\mathcal{G}(\pi(\nu))) = \text{interval}(\pi(\mathcal{G}(\nu)))$ and also $\text{interval}(\text{right}(\mathcal{G}(\nu))) = \text{interval}(\text{right}(\nu))$. Computing 1-interval of $\mathcal{G}(\nu)$ yields $1\text{-interval}(\mathcal{G}(\nu)) = (\alpha, \beta) = (\alpha, \beta + \mathbf{1}\{\beta = n\}) = 1\text{-interval}(\nu)$.

Case 3: $\text{interval}(\nu) = (\epsilon, \delta)$ and $\text{interval}(\text{left}(\nu)) = (\epsilon, \beta)$ for some ϵ and β . Since $\delta < n$, applying Lemma 2, Lemma 3 and Corollary 1 yields $\text{interval}(\mathcal{G}(\nu)) = \text{interval}(\nu)$, $\text{interval}(\mathcal{G}(\pi(\nu))) = \text{interval}(\pi(\mathcal{G}(\nu)))$, and also that $\text{interval}(\text{left}(\mathcal{G}(\nu))) = \text{interval}(\text{left}(\nu))$. Therefore, $1\text{-interval}(\mathcal{G}(\nu)) = (\alpha, \beta) = (\alpha, \beta + \mathbf{1}\{\beta = n\}) = 1\text{-interval}(\nu)$.

Case 4: Because the upperbound of ν is not n , it follows ν is the left child of its parent, and so $(\delta, \epsilon) = \text{interval}(\nu)$ and $(\alpha, \epsilon) = \text{interval}(\text{right}(\nu))$ for some ϵ and α such that $\epsilon < n$. Consequently $1\text{-interval}(\nu) = (\alpha, n)$ for some α and $\text{interval}(\text{right}(\nu)) = (\alpha, \epsilon)$. Applying Lemma 2, Lemma 4 and Corollary 1 we have that $\text{interval}(\mathcal{G}(\nu)) = \text{interval}(\nu)$, $\text{interval}(\pi(\mathcal{G}(\nu))) = (\delta, n + 1)$ and $\text{interval}(\text{right}(\mathcal{G}(\nu))) = \text{interval}(\text{right}(\nu))$ therefore, $1\text{-interval}(\mathcal{G}(\nu)) = (\alpha, n + 1) = (\alpha, \beta + \mathbf{1}\{\beta = n\})$.

Case 5: $\text{interval}(\mathcal{G}(\omega)) = (\gamma, n)$, $\text{interval}(\pi(\mathcal{G}(\omega))) = (\gamma, n + 1)$ and the right child of $\mathcal{G}(\omega)$ is the leaf with label n . Therefore, $1\text{-interval}(\mathcal{G}(\omega)) = (n, n + 1)$.

Case 6: By definition, we have $\pi(\text{left}(\omega)) = \omega$ and so $\text{interval}(\pi(\text{left}(\omega))) = \text{interval}(\omega) = (\gamma, n)$. Because the right child of ω is the leaf with label n it follows that $\text{interval}(\text{left}(\omega)) = (\gamma, n - 1)$. Therefore $\text{interval}(\text{right}(\text{left}(\omega))) = (\alpha, n - 1)$ for some α and $1\text{-interval}(\text{left}(\omega)) = (\alpha, n)$. By Lemma 2 we have $\text{interval}(\mathcal{G}(\text{left}(\omega))) = \text{interval}(\text{left}(\omega))$. By Lemma 5 and Corollary 1 we have that $\text{interval}(\mathcal{G}(\omega)) = \text{interval}(\omega)$. By Lemmas 2 and Lemma 3, we have $\text{interval}(\text{right}(\mathcal{G}(\text{left}(\omega)))) = \text{interval}(\text{right}(\text{left}(\omega)))$, and thus $1\text{-interval}(\mathcal{G}(\text{left}(\omega))) = 1\text{-interval}(\text{left}(\omega))$.

The preceding case analysis computes the 1-interval for every internal node of $\sigma(T)$ that is the image under \mathcal{G} of some node ν in T . In order to complete the 1-intervals($\sigma(T)$) and the proof, we add $1\text{-interval}(\phi)$. □

With these substitution rules for obtaining the intervals and 1-intervals of a tree $\sigma(T)$ from the intervals and 1-intervals of T , we can now proceed to show that if (S, T) is a difficult pair, then so too is $(\sigma(S), \sigma(T))$. We will do so by first showing that the pair $(\sigma(S), \sigma(T))$ do not have a common interval between them and secondly that they have no one-off intervals between them either.

Lemma 7. *Let (S, T) be a difficult pair of extended ordered binary trees of size n , let ω_S be the internal node of S whose right child is the leaf with label n , and let ω_T be the internal node of T whose right child is the leaf with label n . Then the pair of trees $(\sigma(S), \sigma(T))$ do not have an interval in common.*

Proof. Assume, to the contrary, that the pair $(\sigma(S), \sigma(T))$ have an interval in common. Then some interval, call it t , in $\text{intervals}(\sigma(T))$ is also in $\text{intervals}(\sigma(S))$. We consider the possible forms of t given by Lemma 5. If $t = \text{interval}(\omega_T)$, then $t = (\alpha, n)$ and by Lemma 5, $\text{interval}(\omega_S) = t$ which implies (S, T) is not a difficult pair. Otherwise, $t = (\alpha, \delta)$, is some other interval in $\text{intervals}(\sigma(T))$. If $\delta = n + 1$, then the interval (α, n) is in both $\text{intervals}(S)$ and $\text{intervals}(T)$ which implies (S, T) is not a difficult pair. Otherwise, $\delta < n$ and so (α, δ) is in $\text{intervals}(S)$ and $\text{intervals}(T)$ and (S, T) is not a difficult pair. \square

Lemma 8. *Let (S, T) be a difficult pair of extended ordered binary trees of size n , let ω_S be the internal node of S whose right child is the leaf with label n , and let ω_T be the internal node of T whose right child is the leaf with label n . Then the pair of trees $(\sigma(S), \sigma(T))$ have no one-off intervals between them.*

Proof. Assume, to the contrary, that the pair $(\sigma(S), \sigma(T))$ have some one-off interval between them. Without loss of generality, let t be the 1-interval of $\sigma(T)$ that is also an interval of $\sigma(S)$ and consider the possible forms of t given by Lemma 6. By construction, neither $\sigma(S)$ nor $\sigma(T)$ can have the interval $(n, n+1)$ and so $t \neq (n, n+1)$. If $t = 1\text{-interval}(\text{left}(\omega_T))$, then $t = (\lfloor \text{right}(\text{left}(\omega_T)) \rfloor, n) \in \text{intervals}(\sigma(S))$ but then Lemma 5 implies $t = \text{interval}(\omega_S)$ and so (S, T) is not a difficult pair. If $t = 1\text{-interval}(\phi)$ then $t = (\lfloor \pi(\phi_T) \rfloor, n) = (\lfloor \pi(\omega_T) \rfloor, n) = \text{interval}(\pi(\omega_T))$ but then Lemma 5 again implies $t = \text{interval}(\omega_S)$ and so (S, T) is not a difficult pair. If $t = (\alpha, n+1)$, then $(\alpha, n) \in 1\text{-intervals}(T)$ and $(\alpha, n) \in \text{intervals}(S)$ and (S, T) is not a difficult pair. Otherwise, $t = (\alpha, \delta)$ such that $\delta < n$, therefore $(\alpha, \delta) \in 1\text{-intervals}(T)$ and $(\alpha, \delta) \in \text{intervals}(S)$ and so (S, T) is not a difficult pair. \square

We have now established the fact that the pair $(\sigma(S), \sigma(T))$ is a difficult tree pair which underlies the correctness of DPS.

Theorem 1. *Let (S, T) be a difficult pair of extended ordered binary trees of size n , then the pair $(\sigma(S), \sigma(T))$ of extended ordered binary trees of size $n + 1$ is a difficult pair.*

Proof. Immediate from Lemma 7 and Lemma 8. \square

Theorem 2. *Let n be a natural number greater or equal to 4, then the Difficult Pair Sampling algorithm is guaranteed to return a difficult pair of trees of size n .*

Proof. We proceed by induction on n the size of the trees in the difficult pair desired. In the base case, $n = 4$, in which case, DPS samples one of the 4 primitive difficult pairs of trees which have been found by easy enumeration.

Now we suppose that the Difficult Pair Sampling algorithm is guaranteed to return a difficult pair of trees for all $m < n$ such that $m, n \in \mathbb{N}$, $4 < n$, and let (S, T) be a difficult pair of trees of size $n - 1$ sampled by DPS. By Theorem 1 there is at least one pair of difficult trees in the set of all pairs of growth neighbors of S and T which DPS will find by enumeration. Consequently, DPS is guaranteed to return a difficult pair of trees of size n . \square

4 Time Complexity of DPS

We now analyze the time complexity of DPS and show the following:

Theorem 3. *The Difficult Pair Sampling algorithm runs in $O(n^4)$ time, where n is the size of the desired difficult tree pair.*

Proof. Line 1 can be implemented to run in constant time by using a table of the primitive difficult pairs. By returning a pair of pointers line 9 can also be implemented to run in constant time. Therefore the time complexity of DPS is determined by the `for` loop of lines 2 through 8. We name the `for` loops as follows: let f be the `for` loop of lines 2 through 8, g be the `for` loop of lines 4 through 7, and h be the `for` loop of lines 5 through 7.

We consider one iteration of f with (S, T) being the current difficult pair and suppose we use a table, t , to hold the pairs of difficult growth neighbors of (S, T) . Given n , we bound the maximum size of t by the space required for the maximum number of pairs of difficult growth neighbors of size n and so preallocate the space. The space complexity of t is $O(n^3)$. If, on each iteration of f , the candidate difficult growth neighbor pairs are stored from the beginning of the table contiguously, then line 3 can be implemented to run in constant time by starting again at the beginning of the table and keeping track of how many rows have been filled. Further, with such a scheme, line 8 can be implemented to run in $O(n)$ time by randomly selecting a row (in constant time) of one of the candidate difficult pairs and then copying the selected candidates (in linear time) to the space allocated for the current pair. The time complexity of one iteration of f is therefore $O(n + g)$.

The time complexity of one iteration of g is the sum of the time required to compute one growth neighbor of S and the time complexity of h . Using the word representation of an extended ordered binary tree, it is possible to compute a growth neighbor, including its intervals and 1-intervals, in linear time. Thus, one iteration of g takes $O(n + h)$ steps.

The time complexity of one iteration of h is the sum of the time complexity of computing one growth neighbor of T , the time complexity of checking if the resulting pair of growth neighbors, (U, V) , is difficult, and the time complexity of adding (U, V) to *choices*. Now suppose we allocate two, two-dimensional tables, a and b , where we use a to store intervals(S) \cup 1-intervals(T) and b to store intervals(T) \cup 1-intervals(S). These tables will require at most $O(n^2)$ space. If we populate these tables as we construct the growth neighbors, then we can determine whether the pair (U, V) is difficult as we construct V and set a flag appropriately. Then line 6 can be implemented to run in constant time by checking the flag. Assuming the candidate pairs are being stored in table t , then adding the pair (U, V) to *choices* can be a constant time increment operation. So the time complexity of one iteration of h is $O(n)$.

The number of iterations of h is determined by the number of growth neighbors of T . As previously mentioned, for a tree of size n , a straightforward upper bound on the number of growth neighbors is $3n + 1$. Hence, h will execute $O(n)$ times and $O(h) = O(n^2)$. The same reasoning shows that g will also execute $O(n)$ times

and so $O(g) = O(n^3)$. Finally, it is clear that f also executes $O(n)$ times and so $O(f) = O(n^4) = O(\text{DPS})$. \square

5 Sampling coverage of DPS

An ideal sampling algorithm not only has the potential to produce all possible instances of interest, but also produces such instances uniformly at random. One of the excellent features of Remy's algorithm for generating trees is that it selects a given tree uniformly at random from all trees and is ideal for many purposes. This DPS algorithm does not have such uniformity, with some difficult tree pairs being sampled more often than others. Since the number of difficult tree pairs of a particular size is not known exactly, the degree of non-uniformity is difficult to calculate exactly.

From work of Cleary, Elder, Rechnitzer and Taback [2], the fraction of difficult pairs (and in fact its superset, the set of reduced tree pairs) goes to zero exponentially quickly as the size of the tree pairs increase. Calculations by Cleary and Maio [4] show that the number of difficult pairs appear to grow exponentially with an exponential growth rate of about 2.17975, out of the set of equivalence classes of all pairs with an exponential growth rate of about 2.4420, giving a fraction of difficult pairs of less than one in million by size 44 and dropping with increasing size. This is consistent with the observation (proven by Cleary, Rechnitzer and Wong [5]) that for large n , the chance of selecting a difficult tree pair at random is vanishingly small.

As far as the completeness of coverage, for small n where feasible, we found that the DPS algorithm does sample from all hard cases. We considered tens of millions of cases and did not find any difficult instances which were not produced at least once by the DPS algorithm. However, though this gives evidence that the coverage might be complete, this is by no means ensured. There is the potential for there to be difficult tree pairs of size k with no growth neighbors of size $k - 1$. If such difficult pairs existed, they would not be produced by the algorithm ever. We did not encounter any such problematic growth pairs in our computational experiments, but those experiments are necessarily of limited scope and the question remains for later investigation about the completeness of the sampling coverage.

The number of distinct difficult pairs for larger n appears to grow exponentially but the growth is not known exactly and even the exponential rate of growth is not known. The only current means of producing all difficult pairs is equivalent in running time to exhaustive enumeration of all tree pairs and is not feasible beyond small values, so it is not computationally feasible to check to see if instances of even moderate size are not produced by the DPS algorithm. For example, using some natural notions of equivalence described in Cleary and Maio [4], out of 7,152,629,313,600 tree pairs of size 14, there are 17,561,480,528 difficult tree pairs. It is not computationally feasible to test to see if the DPS algorithm will generate all possible instances of that size.

We note that the examples produced by DPS lie in starkly broader classes of

difficult pairs than those specific known earlier examples of Dehornoy [8], Pournin [11], and Cleary and Maio [3]. Those earlier examples give difficult tree pairs of increasingly large sizes but though there are multiple possible examples of increasing size, these numbers do not grow nearly as fast as the set of all possible difficult pairs or as those constructed here. Those examples were constructed for different purposes and there was no intent to get broad coverage of representative difficult instances.

If it is the case that there is complete coverage, the question of the degree of uniformity is of interest. By “degree of uniformity” we mean some indication of the differences in rates of difficult pairs of the same size from being chosen at random. This question has some complications because some of the difficult pairs have certain symmetries arising which raise questions about what exactly is meant in these instances: uniform sampling of instances or of equivalence classes of instances with respect to some natural symmetries arising from the dihedral symmetries of regular polygons. In any case, as far as the degree of uniformity, computations for small n with exhaustive coverage show essentially complete coverage of difficult instances with factors in the range of 2 between the first and third quartiles of number of instances and a factor of about 7 between the most commonly and least commonly generated. There is no reason to presume that this DPS algorithm gives uniformity and these modest experiments show that to be extremely unlikely.

We note that though it is not clear as to the coverage, the DPS algorithm gives a range of difficult pairs. In general, there are many instances where random instances of a problem are by no means representative and often can be attacked with tools that do not capture the essential difficulty. For example, randomly chosen tree pairs do not form good test cases for rotation distance algorithms since with significant probabilities, randomly-selected tree pairs have already many common edges—on average $(16/\pi - 5)n \sim 0.093n$ as proven by Cleary, Rechnitzer, and Wong [5]. Furthermore, randomly selected tree pairs generally have many possible one-off edge reductions as well—also on average $(16/\pi - 5)n \sim 0.093n$ proven by Cleary, Rechnitzer and Wong [5] after compelling numerical evidence of Chu and Cleary [1] estimating those fractions numerically. There are two previously considered algorithms for generating hard cases of increasing size and neither works effectively. The “test and reject” sampling algorithm does give uniform coverage of instances, but the number of needed iterations grows exponentially with size as the scarcity of difficult pairs falls. The “test and reduce to the largest possible difficult subproblem” is not computationally efficient, may not give instances of a prescribed size or even in a prescribed size range, and may well not have uniform coverage of difficult instances. So the DPS algorithm does give a rich range of possible instances in a computationally efficient manner compared to previously known algorithms.

References

- [1] Chu, Timothy and Cleary, Sean. Expected conflicts in pairs of rooted binary trees. *Involve*, 6(3):323–332, 2013. DOI: 10.2140/involve.2013.6.323.

- [2] Cleary, Sean, Elder, Murray, Rechnitzer, Andrew, and Taback, Jennifer. Random subgroups of Thompson's group F . *Groups Geom. Dyn.*, 4(1):91–126, 2010. DOI: 10.4171/GGD/76.
- [3] Cleary, Sean and Maio, Roland. Edge conflicts do not determine geodesics in the associahedron. *SIAM J. Discrete Math.*, 32(2):1003–1015, 2018. DOI: 10.1137/17M1114582.
- [4] Cleary, Sean and Maio, Roland. Counting difficult tree pairs with respect to the rotation distance problem. *J. Combin. Math. Combin. Comput.*, 115:199–213, 2020. DOI: 10.1080/01621459.2018.1537922.
- [5] Cleary, Sean, Rechnitzer, Andrew, and Wong, Thomas. Common edges in rooted trees and polygonal triangulations. *Electron. J. Combin.*, 20(1):Paper 39, 22, 2013. DOI: 10.37236/2541.
- [6] Cleary, Sean and St. John, Katherine. Rotation distance is fixed-parameter tractable. *Inform. Process. Lett.*, 109(16):918–922, 2009. DOI: 10.1016/j.ipl.2009.04.023.
- [7] Culik, II, Karel and Wood, Derick. A note on some tree similarity measures. *Inform. Process. Lett.*, 15(1):39–42, 1982. DOI: 10.1016/0020-0190(82)90083-7.
- [8] Dehornoy, Patrick. On the rotation distance between binary trees. *Adv. Math.*, 223(4):1316–1355, 2010. DOI: 10.1016/j.aim.2009.09.016.
- [9] Hanke, Sabine, Ottmann, Thomas, and Schuierer, Sven. The edge-flipping distance of triangulations. *J.UCS*, 2(8):570–579, 1996.
- [10] Knuth, Donald E. *The art of computer programming. Volume 3*. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont., 1973. Sorting and searching, Addison-Wesley Series in Computer Science and Information Processing.
- [11] Pournin, Lionel. The diameter of associahedra. *Adv. Math.*, 259:13–42, 2014. DOI: 10.1016/j.aim.2014.02.035.
- [12] Rémy, Jean-Luc. Un procédé itératif de dénombrement d'arbres binaires et son application à leur génération aléatoire. *RAIRO Inform. Théor.*, 19(2):179–195, 1985. DOI: 10.1051/ita/1985190201791.
- [13] Sleator, Daniel D., Tarjan, Robert E., and Thurston, William P. Rotation distance, triangulations, and hyperbolic geometry. *J. Amer. Math. Soc.*, 1(3):647–681, 1988. DOI: 10.2307/1990951.

Received: 16th February 2020

Strongly Possible Functional Dependencies for SQL*

Munqath Alattar^{ab} and Attila Sali^{ac}

Abstract

Missing data is a large-scale challenge to research and investigate. It reduces the statistical power and produces negative consequences that may introduce selection bias on the data. Many approaches to handle this problem have been introduced. The main approaches suggested are either missing values to be ignored (removed) or imputed (filled in) with new values [14]. This paper uses the second method. Possible worlds and possible and certain keys were introduced in [22, 25], while certain functional dependencies (*c*-FD) were introduced in [23] as a natural complement to Lien's class of possible functional dependencies (*p*-FD) by [26], and Weak and strong functional dependencies were studied in [25]. The intermediate concept of strongly possible worlds introduced in a preceding paper [3] and results in strongly possible keys (*spKey*'s) and strongly possible functional dependencies (*spFD*'s) were studied. Also, a polynomial algorithm to verify a single *spKey* was given and it was shown that it is NP-complete in general to verify an arbitrary collection of *spKeys*. Furthermore, a graph-theoretical characterization was given for validating a given *spFD* $X \rightarrow_{sp} Y$.

We show, that the complexity to verify a single strongly possible functional dependency is NP-complete in general, then we introduce some cases when verifying a single *spFD* can be done in polynomial time. As a step toward axiomatization of *spFD*'s, the rules given for weak/strong and certain functional dependencies are checked. Appropriate weakenings of those that are not sound for *spFD*'s are listed. The interaction between *spFD*'s and *spKey*'s and certain keys is studied. Furthermore, a graph theoretical characterization of implication between singular attribute *spFD*'s is given.

*Research of the second author was partially supported by the National Research, Development and Innovation Office (NKFIH) grants K-116769, K-132696 and SNN-135643. This work is also supported by the National Research, Development and Innovation Fund (TUDFO/51757/2019-ITM, Thematic Excellence Program), the BME NC TKP2020 grant of NKFIH Hungary and it is also connected to the scientific program of the "Development of quality-oriented and harmonized R+D+I strategy and functional model at BME" project, supported by the New Hungary Development Plan (Project ID: TÁMOP-4.2.1/B-09/1/KMR-2010-0002).

^aDepartment of Computer Science and Information Theory, Budapest University of Technology and Economics, Hungary

^bInformation Technology Research and Development Center, University of Kufa, Iraq, E-mail: m.attar@cs.bme.hu, ORCID: 0000-0002-4673-4902

^cAlfréd Rényi Institute of Mathematics, Budapest, Hungary, E-mail: sali.attila@renyi.hu, ORCID: 0000-0002-4837-6360

Keywords: strongly possible functional dependencies, strongly possible keys
 NULL values, data imputation, matchings in bipartite graphs, list coloring

1 Introduction

Incomplete data in databases are allowed in many of the modern systems. For example, when raw data is collected from different sources in data warehousing, some of the attributes may be available in some of the sources and may not be in some others. For this reason, it is important to treat constraints over incomplete tables. Aliriza et al. [12] showed that encountering up to half of the data values missing is common. So using analysis methods that work only with complete data tables makes mining the data more complicated.

Incompleteness occurs in database tables for different reasons. Date [10] identified seven types of NULL's as follows: value not applicable, value unknown, value does not exist, value undefined, value not valid, value not supplied, and value is the empty set. We consider the second, third, and seventh types as this paper deals with data consumption in an incomplete database table. We consider the symbol N/A for the other types of missing data and we assume it belongs to each domain and it is considered as a regular domain value for comparisons and analysis.

Missing values problem complicates data analysis. It also causes a loss of data efficiency and effectiveness [14]. Although some data analysis approaches get over the problem of incomplete databases, still most approaches require complete databases. To overcome the missing data problem, there are mainly two main methods, either ignoring the incomplete tuples or filling them with domain values chosen by some heuristics. [14].

Possible worlds, introduced by Köhler et al. [22], are obtained by replacing each missing data value with a value from its attribute's domain, which can be infinite. For an incomplete table, every possible world is a complete table that may contain some duplicated tuples. They defined a possible (certain) keys as a key that are satisfied by some (every) possible world of a (non-total) database table. For example, the table in Table 2a satisfies the possible key $\{Course\ Name\}$ as there is a possible world that satisfies it, and every possible world of the table satisfies the certain key $\{Course\ Name, Year, Semester\}$. Furthermore, no possible world of the table satisfies the key $\{Lecturer\}$. Weak (strong) functional dependencies were defined by [25] as FD's that are satisfied by some (every) possible world. The table in Table 1a satisfies the strong FD $\square(Name\ mrg_status \rightarrow gender)$ because every possible world satisfies it, and the weak FD $\diamond(mrg_status \rightarrow gender)$, as there exist some possible worlds, but not all, satisfying the implication.

Most often, especially when the attribute's domain is not known, there is no reason to consider any other attribute value than the already existing values of the table. Types of cars, diagnoses of patients, applied medications, dates of exams, course descriptions, etc, are some examples. For that, the strongly possible world is

Table 1: Possible and Strongly Possible Worlds

(a) Incomplete Table			
Name	gender	mrg_status	age
⊥	female	married	32
Sarah	female	⊥	⊥
David	⊥	divorced	38
James	male	single	⊥
James	male	widower	47

(b) Possible World			
Name	gender	mrg_status	age
30	female	married	32
Sarah	female	lawyer	high
David	Apple	divorced	38
James	male	single	-12
James	male	widower	47

(c) Strongly Possible World			
Name	gender	mrg_status	age
David	female	married	32
Sarah	female	single	32
David	male	divorced	38
James	male	single	38
James	male	widower	47

defined as a possible world achieved by substituting each occurrence of `NULL` with a value from the corresponding attribute's existing values [3]. Strongly possible key(FD) is a key(FD) that satisfied by a strongly possible world. Values that are shown in each attribute of a table represent a part of that attribute's domain. When the attribute domain is unknown, choosing values out of its real domain to fill in a `NULL` may distort the data. For example, it would be unsuitable using value other than the ones appearing in the *marriage status* attribute to fill the \perp in the second row in Table 1a. Other values like numbers, symbols, or any other strings with distant meanings may cause distortion. Therefore, a more meaningful and

semantically acceptable possible (strongly possible) world would be provided using one of the already shown values in the attribute. As the possible world in Table 1c is preferred to the one in Table 1b. Strongly possible worlds were introduced by [3] where strongly possible keys (spKey's) were studied. The natural extension to strongly possible functional dependencies was investigated in [4].

1.1 Contributions

In the present paper, we study spFD's that were earlier introduced in [4, 5] as a generalized version of functional dependency constraint using the visible domain concept and we continue the work started in [3]. We provided a graph-theoretical condition for the spFD satisfaction in [4]. We give a list of axioms of weak/strong and certain FD's that are sound for spFD's, as well and describe several possible sound weakenings for those that are not sound. Interactions between spKeys and strongly possible FDs/certain FDs are investigated, and also the spFD's between singular attributes are studied. The main contributions of this paper are as follows:

- We analysed the properties of strongly possible keys and introduced a (worst-case exponential time) algorithm to verify a single spKey in [3]. A polynomial-time solution for the same problem is given and we show that verifying an arbitrary system of strongly possible keys is NP-complete, thus resolving an open problem from [3].
- A graph-theoretical characterization was given in [4] to verify when spFD $X \rightarrow_{sp} Y$ holds in an SQL table T containing NULL's. We use that here to show that the satisfaction problem for a single spFD $X \rightarrow_{sp} Y$ is NP-complete.
- We give a list of the axioms of Weak/Strong and Certain FD's that are sound for spFD's, with several possible weakenings or restrictions that keep soundness for those that are not sound.
- An interaction between the spKeys and sp/c-FDs is studied and the weakening and transitivity rules are introduced.
- We analyzed the case when the spFD's are restricted to singular attributes. A complete characterisation of the implication problem of this special case is given based on the fact that all the possible extensions of any NULL occurrences are shown in the table.

1.2 Paper structure

The organization of this paper is as follows. Section 2 contains the necessary definitions. Section 3 discusses related work. Algorithmic and complexity questions of strongly possible keys and strongly possible functional dependencies are introduced in Section 4. In section 5, some spFD's implication properties are studied. First the case if one side of an spFD is fixed is treated. Then sound implication rules for

spFD's are listed. The proof of their correctness are given in [5]. Then the special case of spFD's between single attributes is treated using graph theoretical methods. Finally, Section 6 contains concluding remarks and future research directions.

2 Definitions

In this section, we recall some basic definitions introduced in [4]. Let $R = \{A_1, A_2, \dots, A_n\}$ be a relation schema. The set of all the possible values for each attribute $A_i \in R$ is called the domain of A_i and denoted by $D_i = \text{dom}(A_i)$ for $i = 1, 2, \dots, n$.

An instance $T = (t_1, t_2, \dots, t_s)$ over R is a list of tuples that each tuple is a function $t : R \rightarrow \bigcup_{A_i \in R} \text{dom}(A_i)$ and $t[A_i] \in \text{dom}(A_i)$ for all A_i in R . Note that this definition of tuples emphasizes that the order of the attributes in schema R is irrelevant and by taking a list of tuples we use the *bag semantics* that allows several occurrences of the same tuple. For a tuple $t_r \in T$ and $X \subseteq R$, let $t_r[X]$ be the restriction of t_r to X .

It is assumed that \perp is an element of each attribute's domain that denotes missing information. t_r is called V -total for a set V of attributes if $t_r[A] \neq \perp$, $\forall A \in V$. Also a tuple t_r is a total tuple if it is a R -total. t_1 and t_2 are *weakly similar* on $X \subseteq R$ denoted as $t_1[X] \sim_w t_2[X]$ defined by Köhler et al. [22] if:

$$\forall A \in X \quad (t_1[A] = t_2[A] \text{ or } t_1[A] = \perp \text{ or } t_2[A] = \perp).$$

Furthermore, t_1 and t_2 are *strongly similar* on $X \subseteq R$ denoted by $t_1[X] \sim_s t_2[X]$ if:

$$\forall A \in X \quad (t_1[A] = t_2[A] \neq \perp).$$

For the sake of convenience, we write $t_1 \sim_w t_2$ if t_1 and t_2 are weakly similar on R and use the same convenience for strong similarity. Thus, if t_1 and t_2 are both weakly similar to a NULL-free tuple t , then $t_1 \sim_w t_2$. This is true because both of the non-total tuples t_1 and t_2 could be extended to be equal to the total tuple t .

Let $T = (t_1, t_2, \dots, t_s)$ be a table instance over R . $T' = (t'_1, t'_2, \dots, t'_s)$ is a *possible world* of T , if $t_i \sim_w t'_i$ for all $i = 1, 2, \dots, s$ and T' is completely NULL-free. That is, we replace the occurrences of $\perp = t[A_i]$ with a value from the domain D_i different from \perp for all tuples and all attributes.

Weak functional dependency $\diamond X \rightarrow Y$ holds in T if there exists a possible world T' such that $T' \models X \rightarrow Y$ in the classical sense, that is functional dependency $X \rightarrow Y$ holds in T' meaning that if $t'_i[X] = t'_j[X]$ then $t'_i[Y] = t'_j[Y]$ is satisfied, for all pairs of tuples $t'_i, t'_j \in T'$. *Strong functional dependency* $\square X \rightarrow Y$ holds in T if functional dependency $X \rightarrow Y$ holds in all possible worlds T' of T . X is a *possible key* if there exists a possible world T' such that X is a key in T' , and X is a *certain key* if it is a key in every possible world of T . The following was proven in [22].

Theorem 1. $X \subseteq R$ is a certain (possible) key iff

$$\forall t_1, t_2 \in T: t_1[X] \not\sim_w t_2[X] \text{ (} t_1[X] \not\sim_s t_2[X] \text{)}.$$

Table 2: Complete and Incomplete Datasets

(a) Incomplete Dataset

Course Name	Year	Lecturer	Credits	Semester
Mathematics	2019	⊥	5	1
Datamining	2018	Sarah	7	⊥
⊥	2019	Sarah	⊥	2

(b) Complete Dataset

Course Name	Year	Lecturer	Credits	Semester
Mathematics	2019	Sarah	5	1
Datamining	2018	Sarah	7	2
Datamining	2019	Sarah	7	2

2.1 Strongly possible worlds

The concepts of *visible domain* and *strongly possible world* were introduced in [3].

Definition 1. The visible domain of an attribute A_i (VD_i) is the set of all distinct values except \perp that are already used by tuples in T :

$$VD_i = \{t[A_i] : t \in T\} \setminus \{\perp\} \text{ for } A_i \in R$$

For example, the visible domain of the *Credits* attribute in Table 2a is $\{5, 7\}$. So, for any dataset with no information about the attributes' domains, we define their structure and domains by the data itself. It is more reliable and realistic when considering only what information we have in a given dataset and depending on extracting the relationship between data to overcome the missing data problem. Strongly possible worlds are obtained by using the visible domain values in place of the occurrence of NULL's.

Definition 2. A possible world T' of T is called strongly possible world (spWorld) if $t'[A_i] \in VD_i$ for all $t' \in T'$ and $A_i \in R$.

We defined *strongly possible keys* (spKey's) and *strongly possible functional dependencies* (spFD's) in [3, 4] respectively by strongly possible worlds as follows.

Definition 3. Strongly possible functional dependency $X \rightarrow_{sp} Y$ holds in table T over schema R if there exists an spWorld T' of T such that $T' \models X \rightarrow Y$. X is a strongly possible key if there exists an spWorld T' of T such that X is a key in T' , in notation $sp\langle X \rangle$. Note that this is not equivalent to spFD $X \rightarrow_{sp} R$ since we use the bag semantics.

In Table 2a, the spKey of the two attributes {Course Name, Year} is satisfied, and so is the spFD $CourseName \rightarrow_{sp} Semester$, as the strongly possible world in Table 2b shows it.

For a schema R , the *NULL-free subschema (NFS)* is a subset $R_S \subseteq R$ that corresponds to SQL's NOT NULL constraint. A table T over R satisfies NFS R_S if it is R_S -total, that means every tuple $t \in T$ is R_S -total ($\forall A \in R_S: t[A] \neq \perp$). If T satisfies NFS R_S , then we say T is over (R, R_S) . Also, if Σ is a set of integrity constraints (for example spFD's), then a table T over (R, R_S) is an *Armstrong instance* of Σ if

- (1) $T \models \sigma \iff \Sigma \models \sigma$ for any constraint σ , and
- (2) $\forall A \in R \setminus R_S, \exists t \in T$ such that $t[A] = \perp$.

This is the classical Armstrong instance extended by requiring that if an attribute is not in the NULL-free subschema, then it certainly contains NULL's.

3 Related work

Keys and functional dependencies are two main constraints that enforce the semantics of relational databases. Database tables of real database systems usually contain occurrences of null values and for some cases, this happens in candidate key columns. Many studies have been done on handling the missing values problem.

Aliriza et.al. introduced a framework of imputation methods in [14] and evaluated how the selection of different imputation methods affected the performance in [12]. Experimental analyses of several algorithms for imputation of missing values were provided by [16, 13, 2, 7]. An approach introduced by Zhang et.al. discussed and compared different approaches that use only known values [29].

Sree [11] suggests that it is necessary to substitute the missing values with values based on other information in the dataset to overcome the biased effects that affect the accuracy of classification. Likewise, for each NULL in an attribute, we use the attribute's existing values. Cheng et.al.[8] used a clustering algorithm to cluster data and calculate coefficient values between different attributes producing the minimum average error.

Interactions of functional dependencies and other integrity constraints with null values have long been studied. Early investigations focused on "fixing" the database using the Chase procedure, such as Grahne did in [15]. Imlienski and Lipski [19] also investigated the properties of Chase concerning NULL's. Kiss and Márkus [21], provided a chase procedure for functional and inclusion dependencies to provide graph manipulation rules for dependency inference.

The two most used interpretations of NULL's are "value unknown at present" and "no information". The first one leads to possible world semantics that is NULL's are substituted by domain values to obtain total tables. A three-valued model of FD satisfaction is given by Vassilou [27]. It takes into account that all possible words of a table T are considered and a functional dependency either holds, does

not hold or may hold on T . This latter means that in some possible worlds it holds, and in some other ones it does not hold. Fuzzyness is naturally associated with imperfect information. For fuzzy relational databases, the truth value of FD's was studied by Achs and Kiss in [1] Levene and Loizou defined weak and strong functional dependencies based on possible world concept. A weak FD holds in some of the possible worlds and a strong FD holds in all possible worlds. A sound and complete axiom system is given for them in [25].

Both of unknown, as well as non-existing data can be treated by the "no information" approach. Lien [26] defined functional dependencies that hold if strong similarity on the LHS implies equality on RHS. The equality here means that when two tuples are equal on an attribute, if the attribute value is NULL in one tuple, then the other tuple must also be NULL in the same attribute. This dependency is same as the possible FD (p-FD) of Köhler and Link [23]. The novelty of the latter paper is the concept of certain functional dependencies. A c-FD holds if the weak similarity on LHS implies equality on the RHS, and the equality here is defined in the same way as the equality of attribute values on RHS of p-FD's. An axiom system for functional dependencies with NULL's was given by Atzeni and Morfuni [6], but the drawback was that they allowed no NULL's on the LHS.

Levene and Loizou introduced weak and strong FD's, where weak (strong) functional dependency is satisfied if it holds in some (every) possible world [25]. We also used the possible world semantics to introduce our strongly possible functional dependency and it fits between weak and strong FD's. Let suppose we have a table instance with at least one non-NULL value is there in each attribute, then a c-FD $X_w \rightarrow Y$ satisfaction implies the satisfaction of the spFD $X \rightarrow_{sp} Y$. But the satisfaction of $X \rightarrow_{sp} Y$ does not imply the satisfaction of the p-FD $X_s \rightarrow Y$. For example, $Year \rightarrow_{sp} Credits$ is satisfied in Table2a, while $Year_s \rightarrow Credits$ is violated. Example 1 shows a brief comparison of the different functional dependencies. This example is shown in [4], we repeat it here after fixing a typo appearing in the conference version.

Example 1. Let T be the following SQL table.

employee	dept	manager	salary
Knuth	NULL	Chomsky	100,000
Turing	CS	von Neumann	NULL
Turing	NULL	Gödel	NULL

In the following table, we compare the six types of FD's shown in the SQL table: 3-valued [27], weak and strong [25], possible [26], certain [23] and strongly possible functional dependencies (spFD's).

	3-valued	weak	strong	possible	certain	spFD
$e \rightarrow d$	unknown	T	F	F	F	T
$e \rightarrow m$	F	F	F	F	F	F
$e \rightarrow s$	unknown	T	F	T	T	T
$d \rightarrow d$	T	T	T	T	F	T
$d \rightarrow m$	unknown	T	F	T	F	F
$m \rightarrow e$	T	T	T	T	T	T
$m \rightarrow d$	T	T	T	T	T	T

Possible and certain keys were introduced by Köhler et. al. [22], such that a set K of attributes is a possible (certain) key if it is a key in some (every) possible world. The "strongly possible" concept of the present paper is in between of these two, where a strongly possible world is a possible world also. As possible worlds may use any value from an attribute domain to substitute a NULL it may allow an infinite set of values. On the other hand, strongly possible worlds allow only a finite set of values and created from finite attribute domains. Some of the results in [22] assume that some attribute domains are infinite. In this paper, we study the dependencies without that assumption.

Imielinski [18] introduced the construction of OR-relations which is another closely related concept to our spFD's. An OR-tuple over schema $R(A_1, A_2, \dots, A_n)$ is a mapping $t: R \rightarrow \cup_{i=1}^n \mathcal{P}_0(\text{dom}(A_i))$ such that $t[A_i] \in \mathcal{P}_0(\text{dom}(A_i))$, where $\mathcal{P}_0(X)$ is the collection of finite subsets of set X . An OR-relation r is a bag of OR-tuples and a possible world r' for r is a collection of tuples t' such that $t'[A_i] \in t[A_i]$ for all $t \in r$. A wFD $\diamond X \rightarrow Y$ is satisfied by an OR-relation r iff there is a possible world r' of r such that $r' \models X \rightarrow Y$. Hartmann and Link already noted in [17] that Union rule does not hold for wFD's in OR-relations. Our strongly possible world concept can be modelled by OR-relations in such a way that if $t[A_i] = \perp$ for some tuple in an SQL table T , then $t[A_i]$ is replaced by the OR-set $VD(A_i)$, the visible domain of A_i . So strongly possible worlds and spFD's can be considered as special cases of possible worlds of OR-relations and wFD's in them. Our analysis shows, that most of the classical implication rules, axioms do not hold even in this special case.

Finally, Wei and Link [28] aim to define a solid generalization of functional dependencies that do not rely on the interpretation of NULL's. Also, the two papers [23, 28] studied the databases normalization based on the appropriate functional dependencies. It is a future research topic on how our spFD's could be applied for normalization, as normalization is important to eliminate redundancy that may cause inconsistency at updates.

4 Algorithmic and complexity questions

In this section the complexity questions connected to strongly possible keys and strongly possible functional dependencies are studied. The main problems are NP-complete in general, but several important special cases can be solved in polynomial time.

4.1 Strongly possible keys

Following is the algorithmic question we study here. Let T be a SQL table and Σ be a collection of strongly possible key constraints, does $T \models \Sigma$ hold? We may assume, without loss of generality, that there exists at least one spWorld for every treated table. This is a reasonable assumption, as the non-existence of an spWorld happens only if a table contains only NULL's in an attribute.

In [3], an algorithm is given using bipartite matchings for the case when a single spKey needs to be checked, i.e. $\Sigma = \{sp\langle K \rangle\}$. If $K = \{A_1, A_2 \dots A_b\}$, then the running time of that algorithm is $O(|R|(|T| + |T^*|)) + O((|T| + |T^*|)|E|)$, where T^* is the set of total tuples $T^* = \{t^* \in \prod_{i=1}^b VD_i : \exists t \in T \text{ such that } t[K] \sim_w t^*[K]\}$. However, the size of T^* can be an exponential function of size of T . In [4], we gave a polynomial-time refinement of that algorithm. The refined algorithm states that if a single spKey $sp\langle K \rangle$ is to be checked, then considering $T|_K$ is enough, because of K is a key if and only if the tuples are pairwise distinct on K . Next proposition was introduced in [3], it gave a sufficient condition to determine the satisfaction of a given spKey. Let us assume that $T^* \subseteq VD_1 \times VD_2 \times \dots \times VD_b$ is defined by $T^* = \{t^* \in VD_1 \times VD_2 \times \dots \times VD_b : \exists t \in T : t[K] \sim_w t^*\}$, and define a bipartite graph $G = (T, T^*; E)$ with $\{t, t^*\} \in E \iff t[K] \sim_w t^*[K]$. Note that we use here the standard bipartite graph notation, T and T^* are the partite classes. Propositions 1, 2 and 3, Theorem 2, and Algorithm 1 were proven in [4]. We repeat them here to be self-contained.

Proposition 1. $T \models sp\langle K \rangle$ holds if and only if there exists a matching in $G = (T, T^*; E)$ covering T .

We may resort to generating only part of T^* to ensure that the algorithm will run in polynomial time. Let $T = \{t_1, t_2 \dots t_m\}$ and $\ell(t_i) = |\{t^* \in VD_1 \times VD_2 \times \dots \times VD_b : t^* \sim_w t_i[K]\}|$. Note that $\ell(t_i) = \prod_{j: t_i[A_j] \neq \perp} |VD_j|$, then these values can be calculated by scanning T once and using appropriate search tree data structures to hold values of visible domains of each attribute. Sort tuples of T in non-decreasing $\ell(t_i)$ order, i.e. assume that $\ell(t_1) \leq \ell(t_2) \leq \dots \leq \ell(t_p)$. Let $j = \max\{i : \ell(t_i) < i\}$ and $T_j = \{t_1, t_2, \dots, t_j\}$, and furthermore, $T_j^* = \{t^* : \exists t \in T_j : t^* \sim_w t[K]\} \subseteq VD_1 \times VD_2 \times \dots \times VD_b$. Note that $|T_j^*| \leq \frac{1}{2}j(j-1)$. If $\forall i = 1, 2, \dots, m : \ell(t_i) \geq i$, then define $j = 0$ and $T_j^* = \emptyset$.

Proposition 2. $T \models sp\langle K \rangle$ holds if and only if $j = 0$ or there exists a matching in $G' = (T_j, T_j^*; E|_{T_j \times T_j^*})$ covering T_j .

Proposition 2 was proven in [4] and it gives the basis of a polynomial-time algorithm to determine whether $T \models sp\langle K \rangle$ holds or not.

Algorithm 1 Verifying $T \models sp\langle K \rangle$

Input: Table T over schema R , $K \subseteq R$

Output: Strongly possible world T^* showing $T \models sp\langle K \rangle$ if exists

```

1: procedure spKey(item  $T$ , item  $R$ , item  $K$ )
2:   Calculate  $\ell(t): t \in T$ 
3:   Sort  $T_i$  in non-decreasing  $\ell(t_i)$  order
4:    $j \leftarrow \max\{i: \ell(t_i) < i\}$ 
5:   Construct bipartite graph  $G' = (T_j, T_j^*; E|_{T_j \times T_j^*})$ 
6:    $M = \mathbf{MaxMatching}(G')$ 
7:   if  $|M| < j$  then return  $T \not\models sp\langle K \rangle$ 
8:      $T^* \leftarrow M \cap T_j^*$ 
9:     for  $k = j + 1$  to  $|T|$  do
10:      Generate  $t_k^* \notin T^*$  such that  $t_k \sim_w t_k^*$ 
11:       $T^* \leftarrow T^* \cup \{t_k^*\}$ 
12:     end for
13:     return  $T^*$ 
14:   end if
15: end procedure

```

Algorithm 1 was introduced in [4] and its running time is $O(|K| \cdot |T| \log |T| + |T|^5)$. Theorem 2 shows that the spKey problem is NP-complete in general, and its proof is in [4]. We reduced the problem of deciding whether $T \models \Sigma$ for a collection Σ of spKey constraints to the problem of finding the maximal common independent set of three or more matroids in [3]. This matroid intersection problem is NP-complete, however the reduction given was not one-to-one, so it did not prove, just hinted the NP-completeness of spKey problem. Finally, by modifying an argument of [22], we could prove the following theorem in [4] by a Karp-reduction of 3SAT to our problem.

Theorem 2. *The strongly possible key satisfaction problem is NP-complete.*

In some special cases, $T \models \Sigma$ can be verified in polynomial time, where Σ is a collection of strongly possible key constraints, as the following Proposition shows.

Proposition 3. *Let us assume that T is a table over schema R , furthermore let $\Sigma = \{sp\langle K_1 \rangle, sp\langle K_2 \rangle, \dots, sp\langle K_m \rangle\}$ be a collection of spKey constraints. If $K_i \cap K_j = \emptyset$ for $i \neq j$, then $T \models \Sigma$ can be decided in polynomial time.*

4.2 Graph theoretical characterization of strongly possible functional dependencies

In this section, we introduce a graph-theoretical characterization that determines when $T \models X \rightarrow_{sp} Y$. Recall that for a table T over Schema R $T \models X \rightarrow_{sp} Y$ if and only if there exists an spWorld T' of T such that $T' \models X \rightarrow Y$. If $T = \{t_1, t_2, \dots, t_p\}$ and $T' = \{t'_1, t'_2, \dots, t'_p\}$ with $t_i \sim_w t'_i$, then t'_i is called an *sp-extension* or in short an *extension* of t_i . Let $X \subseteq R$ be a set of attributes and let $t_i \sim_w t'_i$ such that for each $A \in R$: $t'_i[A] \in VD(A)$, then $t'_i[X]$ is a *strongly possible extension of t_i on X* . We exclude any attribute with all NULL values, because the visible domain for such attribute is the empty set. We recall the *weak similarity graph* [4], as a useful tool in investigations of strongly possible functional dependencies (spFD's in short).

Definition 4. Let $T = \{t_1, t_2, \dots, t_p\}$ be a table (instance) over schema R . The weak similarity graph G_Y with respect to $Y \subseteq R$ is defined as $G_Y = (T, E)$, where $\{t_i, t_j\} \in E \iff t_i[Y] \sim_w t_j[Y]$.

Theorem 3 characterizes using weak similarity graphs when $T \models X \rightarrow_{sp} Y$ holds. The theorem was proved in [4].

If $T \models X \rightarrow_{sp} Y$, then there exists an spWorld T' of T such that $T' \models X \rightarrow Y$. This latter one holds iff whenever $t'_1[Y] \neq t'_2[Y]$ then $t'_1[X] \neq t'_2[X]$ is also satisfied. Now, if $t_1, t_2 \in T$ are such tuples that $t_1[Y] \not\sim_w t_2[Y]$, then certainly $t'_1[Y] \neq t'_2[Y]$ holds in any spWorld T' . That is, in order to $T \models X \rightarrow_{sp} Y$ hold t'_1, t'_2 must also satisfy $t'_1[X] \neq t'_2[X]$. Recall that by Definition 4, $t_1[Y] \not\sim_w t_2[Y]$ holds exactly when $\{t_1, t_2\}$ is an edge in the *complement* of the weak similarity graph G_Y . We may think about $t'_i[X]$'s as *colors* assigned to vertices of $\overline{G_Y}$ and then we obtain that this coloring must be proper. Now colors that can be assigned to vertices come from lists special to vertices, namely from the sets *strongly possible extensions of t on X* . Let T be a table over schema R , $t \in T$ and $X \subseteq R$. t' is a *strongly possible extension of t on X* if $t[X] \sim_w t'[X]$, t' is X -total and $\forall X_i \in X: t'[X_i] \in VD_{X_i}$. The following theorem introduced in [4] characterizes when $T \models X \rightarrow_{sp} Y$ holds using weak similarity graphs. It tells us that the existence of proper coloring of $\overline{G_Y}$ using the lists determined by the strongly possible extensions on X is a necessary and sufficient condition for $T \models X \rightarrow_{sp} Y$ to hold.

Let $G(V, E)$ be a graph and $L: V \rightarrow 2^{\mathbb{N}}$ be a mapping that assigns each vertex a *list of colors* $L(v)$. A *list coloring of G using lists $\{L(v): v \in V\}$* is a mapping $c: V \rightarrow \bigcup_{v \in V} L(v)$ such that $c(v) \in L(v)$ and $c(u) \neq c(v)$ if $\{u, v\} \in E$. We use Theorem 3 in proofs of sound inference rules.

Theorem 3. Let $T = \{t_1, t_2, \dots, t_m\}$ be a table over schema R . For $X, Y \subseteq R$, $T \models X \rightarrow_{sp} Y$ holds iff $\overline{G_Y}$ can be list colored using lists $\{t_i^1, t_i^2, \dots, t_i^{r_i}\}$ for $t_i \in T$, where $\overline{G_Y}$ is the complement of the weak similarity graph on Y and t_i^j 's are the *strongly possible extensions of t_i on X* .

Table 3 illustrates that sets of tuples that are pairwise weakly similar on Y form a weak similarity clique. Now, tuples from a given clique have a unique non-NULL

value in each attribute of Y , unless they all contain NULL in that attribute. In both cases, there is a way to extend each tuple that tuples in the same clique become identical on Y . So those tuples that are in one weak similarity clique on Y can be list colored by the same color on X so that $T \models X \rightarrow_{sp} Y$. Note that weak similarity cliques of G_Y are independent vertex sets in $\overline{G_Y}$.

Table 3: Color classes and weak similarity cliques.

X	Y
$\left[\begin{array}{c} \text{Same color} \end{array} \right]$	$\left[\begin{array}{c} \text{Weak Similarity Clique} \end{array} \right]$
$\left[\begin{array}{c} \text{Same color} \end{array} \right]$	$\left[\begin{array}{c} \text{Weak Similarity Clique} \end{array} \right]$
$\left[\begin{array}{c} \text{Same color} \end{array} \right]$	$\left[\begin{array}{c} \text{Weak Similarity Clique} \end{array} \right]$

4.3 Complexity of strongly possible functional dependencies

List coloring problem is NP-complete even if all lists have length three [24]. This suggests that deciding whether a given spFD is satisfied in an SQL table is NP-complete even if all the tuples have a maximum of three extensions. However, it is not obvious the "intractable cases" of list coloring problem really correspond to spFD-satisfaction, so we give a direct proof of NP-completeness.

Definition 5. *The SPFD-SATISFACTION problem is defined as follows.*

Input: *An SQL-table T over schema R , $X, Y \subseteq R$.*

Question: *Does $T \models X \rightarrow_{sp} Y$ hold?*

Theorem 4. *The SPFD-SATISFACTION problem is NP-complete.*

Proof. SPFD-SATISFACTION is in NP, since an spWorld T' of T such that $T' \models X \rightarrow Y$ is a good witness. It is clearly of polynomial size of the input and whether $T' \models X \rightarrow Y$ holds can be checked in polynomial time by pairwise comparisons of tuples.

In order to prove that SPFD-SATISFACTION is NP-hard a Karp-reduction from 3-COLOR to SPFD-SATISFACTION is given. Let $G = (V, E)$ be an input of 3-COLOR with $V = \{v_1, v_2, \dots, v_n\}$. An SQL table T is constructed over schema $R = \{A_0, A_1, \dots, A_n\}$ of $n + 1$ tuples and $X = \{A_0\}$ and $Y = \{A_1, A_2, \dots, A_n\}$ is set so that the complement $\overline{G_Y}$ of the weak similarity graph on Y is isomorphic

to G plus three isolated vertices. Let t_1, t_2, \dots, t_n be defined by induction on i as follows. $t_1[A_1] = 1$. If t_1, t_2, \dots, t_i are defined for A_1, A_2, \dots, A_i , then let

$$t_{i+1}[A_j] = \begin{cases} \perp & \text{if } 1 \leq j \leq i \\ 1 & \text{if } j = i + 1 \end{cases}$$

furthermore, for $j = 1, 2, \dots, i$

$$t_j[A_{i+1}] = \begin{cases} 2 & \text{if } \{v_j, v_{i+1}\} \in E(G) \\ \perp & \text{if } \{v_j, v_{i+1}\} \notin E(G) \end{cases}$$

Finally, let t_{n+k} be defined for $k = 1, 2, 3$ as $t_{n+k}[A_0] = k$, $t_{n+k}[A_j] = \perp$ for $j = 1, 2, \dots, n$. Obviously, T can be constructed from G in polynomial time. Our claim is that G is 3-colorable iff $T \models X \rightarrow_{sp} Y$. Indeed, t_{n+k} for $k = 1, 2, 3$ are isolated vertices in $\overline{G_Y}$, so $\overline{G_Y}$ is list-colorable with extensions of t_j over X iff $\overline{G_Y}$ restricted to $W = \{t_1, t_2, \dots, t_n\}$ is list-colorable. Observe that $\overline{G_Y}|_W$ is isomorphic to $G = (V, E)$, Indeed, for $1 \leq i < j \leq n$ $t_i[Y] \not\sim_w t_j[Y]$ if $\{v_i, v_j\} \in E(G)$, because $t_i[A_j] = 2$ and $t_j[A_j] = 1$, on the other hand if $\{v_i, v_j\} \notin E(G)$, then $t_i[A_j] = \perp$ and $t_j[A_j] = 1$, $t_i[A_i] = 1$ and $t_j[A_i] = \perp$, furthermore $t_i[A_\ell], t_j[A_\ell] \in \{\perp, 2\}$ for $\ell \notin \{i, j\}$. Finally, $VD_X = \{1, 2, 3\}$, so the list of extensions of t_i on X is $\{1, 2, 3\}$ for all $1 \leq i \leq n+3$. Thus any proper list coloring of $\overline{G_Y}$ with extensions of t_i on X gives a 3-coloring of G , and vice versa, any 3-coloring of G can easily be extended to a proper list coloring of $\overline{G_Y}$ with extensions of t_i . \square

However, this problem can be solved in polynomial time for some special cases. Complete graphs can be reduced to the verifying spKey problem, as the following proposition shows.

Proposition 4. *$T \models X \rightarrow_{sp} Y$ can be decided in polynomial time if $\overline{G_Y}$ is a complete graph.*

Proof. As $\overline{G_Y}$ is a complete graph, then the tuples are pairwise distinct by some non-NULL value on Y , then, all the tuples in T should be pairwise distinctly colored on X to satisfy the spFD $X \rightarrow_{sp} Y$. That is, $T \models X \rightarrow_{sp} Y \iff T \models sp\langle X \rangle$ that can be checked in polynomial time using Algorithm 1. \square

Greedy algorithm can be applied to find a proper list coloring of a graph $G = (V, E)$ if for $\forall v \in V$ the list size of v is at least $d_G(v) + 1$. This can naturally be applied in our context, as well, that is $T \models X \rightarrow_{sp} Y$ can be decided in polynomial time if $\forall t \in T$, number of all extensions of t on X is larger than the number of weakly similar tuples to t on Y .

In order to apply list coloring, first the lists should be generated. However, the number of X -extensions of t of a tuple $t \in T$ may easily be exponential function of the size of the input as it was seen in Section4.1. This obstacle can be overcome by the following observation since it is enough to generate at most $\Delta(\overline{G_Y}) + 1$ X -extensions for each tuple t .

Proposition 5. *There exists a list coloring with the lists generated for each tuple of size less than or equal $\Delta(\overline{G_Y}) + 1$ if and only if there exists list coloring with full lists.*

Proof. \Rightarrow Indeed, same list coloring can be used.

\Leftarrow Assume there exists list coloring with full lists. And let $\{t_1, t_2, \dots, t_r\}$ be set of tuples with number of extensions on X at most $\Delta(\overline{G_Y})$ and let $\{t_{r+1}, t_{r+2}, \dots, t_s\}$ be the set of other tuples. Take $\Delta(\overline{G_Y}) + 1$ elements lists for $\{t_{r+1}, t_{r+2}, \dots, t_s\}$. Then, keep the coloring of $\{t_1, t_2, \dots, t_r\}$ and then color $\{t_{r+1}, t_{r+2}, \dots, t_s\}$ using greedy algorithm. \square

If $\overline{G_Y}$ is a tree, the list coloring problem can be solved using dynamic programming techniques in polynomial time [20]. Generally, list coloring problem restricted to graphs of maximum degree two, such as cyclic graphs, is polynomially solvable [24].

Weak similarity graphs can be used on the LHS of an spFD in a special case. Namely, if components of weak similarity graph G_X with respect to X are cycles of length at least 4, then $T \models sp\langle X \rangle$, in particular for any $Y \subseteq R$, $T \models X \rightarrow_{sp} Y$ holds.

Proposition 6. *If each connected component C of the weak similarity graph G_X with respect to X is a cycle of length ≥ 4 , then $sp\langle X \rangle$ holds.*

Proof. Let $T = \{t_1, t_2, \dots, t_p\}$ be a table over schema R such that each component C of the weak similarity graph G_X with respect to X is a cycle of length ≥ 4 . We need NULL-free tuples from $t'_i \in VD_1 \times VD_2 \times \dots \times VD_n$ such that $t_i[X] \sim_w t'_i[X]$ and $t'_i[X]$'s are pairwise distinct in order to $sp\langle X \rangle$ hold. Since two tuples can only have identical extensions on X if they are weakly similar on X , it is enough to construct $t'_i[X]$'s for each component of G_X separately. If this single component is a circle $(t_1[X] \sim_w t_2[X] \sim_w t_3[X] \sim_w \dots \sim_w t_k[X] \sim_w t_1[X])$, then any extension of t_2 is distinct on X from any extensions of $t_4 \dots t_k$. There exist $A \in X$ such that $t_1[A] \neq t_3[A]$ and both $t_1[A], t_3[A] \neq \perp$, because $t_1[X] \not\sim_w t_3[X]$. We need to make t'_2 different from t'_1 and t'_3 , so that we can set $t'_2[A] = t_3[A]$ and this distinguishes it from t'_1 . Applying the same idea around the cycle, $t_i[X] \sim_w t_{i+1}[X] \sim_w t_{i+2}[X]$ will make t'_i and t'_{i+1} distinct. \square

5 Implications among strongly possible functional dependencies

This section treats the implication properties of spFD's. The cases when one side of the dependency is a fixed attribute set is characterized, then axioms and rules of weak, strong, possible and certain FD's are analysed with respect to spFD's. In addition to that, the interactions with different key concepts are treated. Finally, a complete characterization of the singular attribute case is given.

5.1 Strongly possible functional dependencies with one side fixed

For $T \models X \rightarrow_{sp} Y$, fixing the left-hand side of the spFD, makes the right-hand sides form a down-set, i.e, if $Y' \subset Y$, then $T \models X \rightarrow_{sp} Y'$ also holds. We introduced Proposition 7 in a proceedings paper [4], we repeat it here to fix an error in the proof of that version. It shows that for fixed left-hand side of spFD's, there is no other restriction for the right-hand sides than forming a down-set.

Proposition 7. *Let (R, R_S) be a schema and $X \not\subseteq R_S$. Let \mathcal{Y} be a down-set of subsets of $R \setminus X$. Then there exists a table T over (R, R_S) such that $T \models X \rightarrow_{sp} Y$ holds iff $Y \cap (R \setminus X) \in \mathcal{Y}$.*


Proof. Let the maximal elements of \mathcal{Y} be Y_1, Y_2, \dots, Y_s , that is $\mathcal{Y} = \{A : \exists i A \subseteq Y_i\}$ and $Y_i \not\subseteq Y_j$ for $i \neq j$. Let $A_0 \in X \setminus R_S$ be a fixed attribute, and A_1, A_2, \dots, A_n be the other attributes of R . Table T contains tuples t_0, t_1, \dots, t_s such that

$$t_0[A_i] = \begin{cases} \perp & \text{if } i = 0 \\ 0 & \text{if } i > 0 \end{cases}$$

and

$$t_i[A_j] = \begin{cases} 0 & \text{if } A_j \in Y_i \cup X \\ i & \text{if } A_j \notin Y_i \cup X \end{cases} \quad \text{for } i = 1, 2, \dots, s \text{ and } j = 1, 2, \dots, n$$

finally $t_i[A_0] = i$ for $i = 1, 2, \dots, s$. So, the table is constructed as follows.

A_0	A_1	\dots	A_n
\perp	0	\dots	0
i	0	\dots	i
			
$Y_i \cup X$			

$Y \in \mathcal{Y} \iff \exists 1 \leq i_Y \leq s : Y \subseteq Y_{i_Y}$, so FD $X \rightarrow Y$ holds in the spWorld obtained by replacing \perp in $t_0[A_0]$ by i_Y , because only tuples t_0 and t_{i_Y} can agree in X . On the other hand, if $Y \notin \mathcal{Y}$, then for every $1 \leq i \leq s$ there exists an attribute $A_{j_i} \in Y \setminus Y_i$, so whichever element $i \in VD_{A_0}$ is put in place of \perp in $t_0[A_0]$, we get that $t_0[X] = t_i[X]$, but $t_0[Y] \neq t_i[Y]$, hence $T \not\models X \rightarrow_{sp} Y$. \square

We can characterize the case of fixed right-hand side of an spFD as well, as it is clear that for a fixed set $Y \subset R$, the collection of attribute sets $\mathcal{X} = \{X : T \models X \rightarrow_{sp} Y\}$ forms an up-set, where T is a table over a schema R . However, this condition is the only one we have. This is shown in Proposition 8 and Theorem 5 proved in [4]. We repeat them here for the sake of completeness.

Proposition 8. *Let (R, R_S) be a schema, $Y \subseteq R$ be a fixed set of attributes, furthermore let \mathcal{X} be an upset of subsets of $R \setminus Y$. Then there exists a table T over (R, R_S) such that $T \models X \rightarrow_{sp} Y \iff X \in \mathcal{X}$.*

The proof uses the Armstrong instance construction for strongly possible keys from [3].

Theorem 5. *Suppose that $\Sigma = \{sp\langle K \rangle : K \in \mathcal{K}\}$ is a collection of spKey constraints such that if $|K| = 1$, then $K \subseteq R_S$. Then, there exists an Armstrong table for (R, R_S, Σ) .*

5.2 Strongly possible functional dependencies axiomatisation

The first steps towards a possible axiomatisation of spFD’s were given in [5]. We studied and analyzed the axioms of weak/strong FD’s given by Levene and Loizou [25] and also the axioms of certain FD’s given by Köhler and Link [23] in context of spFD’s. The investigation showed that some of these axioms are not sound for spFD’s. Table 4 shows the axioms that are still sound for spFD’s and it also shows several possible weakenings and restrictions that keep soundness for those that are not. All proofs and counterexamples are detailed in [4, 5]. We repeat them in this paper to be self-contained.

We may conclude from Table 4 that more than one spFD in the premise of a rule usually cause problems in soundness. This is caused by the fact that a single spWorld may not satisfy the different spFD’s due to the limitations of visible domains. This problem must be handled for a complete axiomatization. Particularly, the fact that composition rule is not sound in general makes usual proof methods of completeness virtually unusable.

Table 4: spFD Axioms Soundness

Axiom	spFD Soundness
Reflexivity	Sound: If $Y \subseteq X \subseteq R$ then $T \models X \rightarrow_{sp} Y$
Augmentation	Sound: If $T \models X \rightarrow_{sp} Y$ and $W \subseteq R$, then $T \models XW \rightarrow_{sp} YW$
Union	<p>Not Sound: If $T \models X \rightarrow_{sp} Y$ and $T \models X \rightarrow_{sp} Z$, then $T \models X \rightarrow_{sp} YZ$</p> <p>Possible weakenings are:</p> <ul style="list-style-type: none"> • Strong FD Mixed-union: If $T \models \square X \rightarrow Y$ and $T \models X \rightarrow_{sp} Z$, then $T \models X \rightarrow_{sp} YZ$ or if $T \models X \rightarrow_{sp} Y$ and $T \models \square X \rightarrow Z$, then $T \models X \rightarrow_{sp} YZ$ • Certain FD Mixed-union: If $T \models X \rightarrow_{sp} Y$ and $T \models X_w \rightarrow Z$, then $T \models X \rightarrow_{sp} YZ$ • NULL-free union: If $T \models X \rightarrow_{sp} Y$ and $T \models X \rightarrow_{sp} Z$ and $X \subseteq R_S$, then $T \models X \rightarrow_{sp} YZ$

Transitivity	<p>Not Sound: If $T \models X \rightarrow_{sp} Y$ and $T \models Y \rightarrow_{sp} Z$, then $T \models X \rightarrow_{sp} Z$</p> <p>Possible weakenings are:</p> <ul style="list-style-type: none"> • Strong FD Mixed-transitivity: If $T \models X \square \rightarrow Y$ and $T \models Y \rightarrow_{sp} Z$ or if $T \models X \rightarrow_{sp} Y$ and $T \models \square Y \rightarrow Z$, then $T \models X \rightarrow_{sp} Z$. • Certain FD Mixed-transitivity: If $T \models X \rightarrow_{sp} Y$ and $T \models Y \rightarrow_w Z$, then $T \models X \rightarrow_{sp} Z$ • Sp-transitivity: If $T \models X \rightarrow_{sp} Y$ and $T \models Y \rightarrow_{sp} Z$, and $Y \subseteq R_s$, then $T \models X \rightarrow_{sp} Z$
Pseudo-transitivity	<p>Not Sound: If $T \models X \rightarrow_{sp} Y$ and $T \models YZ \rightarrow_{sp} V$, then $T \models XZ \rightarrow_{sp} V$</p> <p>or if $T \models X \rightarrow_{sp} YZ$ and $T \models Y \rightarrow_{sp} V$, then $T \models X \rightarrow_{sp} ZV$</p> <p>Possible weakenings are:</p> <ul style="list-style-type: none"> • Strong FD Mixed-pseudo-transitivity: If any of the given spFD's is strong FD, it is sound. • Certain FD Mixed-pseudo-transitivity I: If $T \models X \rightarrow_{sp} Y$ and $T \models XY \rightarrow_w Z$, then $T \models X \rightarrow_{sp} Z$ • Certain FD Mixed-pseudo-transitivity II: If $T \models X \rightarrow_{sp} Y$ and $T \models YZ \rightarrow_w V$, then $T \models XZ \rightarrow_{sp} V$ • NULL-free pseudo-transitivity: If $T \models X \rightarrow_{sp} YZ$ and $T \models Y \rightarrow_{sp} V$ and $Y \subseteq R_S$, then $T \models X \rightarrow_{sp} ZV$.
Composition	<p>Not Sound: If $T \models X \rightarrow_{sp} Y$ and $T \models A \rightarrow_{sp} B$, then $T \models XA \rightarrow_{sp} YB$</p> <p>Possible weakenings are:</p> <ul style="list-style-type: none"> • Mixed composition: If $T \models \square X \rightarrow Y$ and $T \models A \rightarrow_{sp} B$, or if $T \models X \rightarrow_{sp} Y$ and $T \models \square A \rightarrow B$, then $T \models XA \rightarrow_{sp} YB$. • NULL-free composition: If $T \models X \rightarrow_{sp} Y$ and $T \models A \rightarrow_{sp} B$ and $YA \subseteq R_S$, then $T \models XA \rightarrow_{sp} YB$. • Disjoint composition: If $T \models X \rightarrow_{sp} Y$ and $T \models A \rightarrow_{sp} B$ and $X \cap A = \emptyset$, then $T \models XA \rightarrow_{sp} YB$.
Decomposition	<p>Sound: If $T \models X \rightarrow_{sp} YZ$, then $T \models X \rightarrow_{sp} Y$ and $T \models X \rightarrow_{sp} Z$.</p>

5.2.1 sp-keys and sp-FD/c-FDs interaction

For a relation R , the ordinary functional dependency $X \rightarrow R$ implies that X is a key in R , because duplicate rows are prohibited in the relational model [9]. On the other hand, for an SQL table T , the spFD $X \rightarrow_{sp} T$ does not imply the spKey $sp\langle X \rangle$, because duplicate rows are permitted in the bag semantics of data in SQL tables.

We study interaction between sp-keys, certain keys, spFD's and c-FDs in the following.

- **sp-key/spFD Weakening:** $\frac{sp\langle X \rangle}{(X \rightarrow_{sp} Y)} \forall Y \subseteq R$
 Indeed, if there exists an spWorld such that all the tuples are pairwise distinct on X , then $X \rightarrow R$ holds in that spWorld.
- **certain key/spFD Transitivity:** $\frac{(X \rightarrow_{sp} Y) c\langle XY \rangle}{sp\langle X \rangle}$
 The certain key $c\langle XY \rangle$ implies that all the tuples are not pairwise weakly similar on XY . Then, for every two tuples, they either have distinct and non-NULL values on X or on Y . But we have $X \rightarrow_{sp} Y$, then \overline{G}_Y can be list colored by X extensions, and we can extend the extensions of this coloring in any way to R . So, by using this coloring, $sp\langle X \rangle$ is satisfied.
 However, the sp-key/FD Transitivity true in following form is not sound:
 $\frac{(X \rightarrow_{sp} Y) sp\langle XY \rangle}{sp\langle X \rangle}$. The reason is the fact that there will be a possibility of encountering a weak similarity on X and distinctness on Y , as illustrated in Table 5.

Table 5: sp-key/spFD Transitivity dissatisfaction

	X	Y
1	1	1
1	⊥	1
1	2	2

5.3 spFDs for singular attributes

Substituting a value from the visible domain in place of a \perp produces a duplication in that attribute, since for a singular attribute, the visible domain represents all the possible replacements for any \perp occurrence. For example, in Table 6, all possible substitutions of the NULL in the second tuple on X are $\{1, 2, 3\}$, as they form VD_A . A singular attribute can only be an sp-key if there are no \perp 's in that attribute. However, spFD's are possible between singular attributes with NULL's. Any occurrence of a NULL in the LHS of an spFD for a singular attribute causes a duplication, and this requires a corresponding duplication possibility on the RHS to satisfy the spFD. In the example in Table 6, $X \rightarrow_{sp} Y$ holds because, in the first and the third tuples, there is a duplication possibility on Y for the NULL replacement on X in the second tuple. On the other hand, $Y \not\rightarrow_{sp} X$ holds, because any replacement for the NULL in the third tuple on Y will not get a corresponding duplication possibility on X .

In the present subsection, we study the case of singular attributes as a special case of spFD implication. The following proposition shows a bidirectional property for singular attributes spFD.

Table 6: spFD for Singular Attributes

X	Y
1	1
\perp	1
2	\perp
3	2

Proposition 9. For singular attributes X and Y , if $T \models X \rightarrow_{sp} Y$ and $|VD_X| = |VD_Y|$, then $T \models Y \rightarrow_{sp} X$

Proof. We may assume without loss of generality that $VD_X = \{1, 2, \dots, \ell\}$. Since $T \models X \rightarrow_{sp} Y$, for every $i \in \{1, 2, \dots, \ell\}$ if $t \in T$ is a tuple, then $t[X] = i$ implies that $t[Y] = a_i$ or $t[Y] = \perp$. Also, let $\{b_1, b_2, \dots, b_r\} \subset VD_Y$ be those pairwise distinct visible domain values that satisfy $t[Y] = b_j \Rightarrow t[X] = \perp$. Assume that for $1 \leq i \leq k$ we have non-NULL a_i such that there exists a tuple t with $t[X] = i$ and $t[Y] = a_i$, that is, for $k+1 \leq j \leq \ell$ we have $t[X] = j \Rightarrow t[Y] = \perp$. Table 7 shows the possible types of tuples on $\{X, Y\}$. If $a_i \neq a_j$ for $i \neq j$, then by $|VD_X| = |VD_Y|$ we have that $r = \ell - k$ and the NULL's can easily be substituted in these two columns so that $1 \leq i \leq k$ is matched with a_i and $k+1 \leq j \leq \ell$ is matched with b_{j-k} so the two columns are basically identical, that is $Y \rightarrow_{sp} X$ holds. On the other hand, if $a_i = a_j$ for some $i \neq j$, then we have that $r > \ell - k$ since the sizes of divisible domains of X and Y are the same. Consider a spWorld T' of T that satisfies $T' \models X \rightarrow Y$ functional dependency. Such a T' exists, since $T \models X \rightarrow_{sp} Y$. Now, the NULL's in the tuples that have value b_j in Y can only be substituted by values from $\{k+1, \dots, \ell\}$, because for $1 \leq i \leq k$ there are tuples with non-NULL values $t[X] = i$ and $t[Y] = a_i$. However, as $r > \ell - k$, we must have $1 \leq u < v \leq r$ and $k+1 \leq j \leq \ell$ so that we have two tuples t, t' with $t[X] = t'[X] = j$ and $t[Y] = b_u$ and $t'[Y] = b_v$ contradicting to $T' \models X \rightarrow Y$. \square

Example 2. The table below shows an instance T with $T \models X \rightarrow_{sp} Y$ and $|VD_X| = |VD_Y| = 2$. Value 2 in attribute X is shown only in t_4 where $t_4[Y] = \perp$. Then, to have $|VD_X| = |VD_Y|$, we have value a_2 in attribute Y is shown only in tuples having \perp on X . Then, X and Y are two singular attributes satisfy a bidirectional implication property.

Proposition 10. For singular X and Y , if $T \models X \rightarrow_{sp} Y$, then $|VD_X| \geq |VD_Y|$.

Proof. Let T' be a spWorld of T that satisfies functional dependency $X \rightarrow Y$ that exists by $T \models X \rightarrow_{sp} Y$. Observe that the set of values appearing in column X of T' is VD_X , while that of values appearing in column Y is VD_Y . Since if $t'_1[X] = t'_2[X]$ implies $t'_1[Y] = t'_2[Y]$ for any $t'_1, t'_2 \in T'$, the mapping $f: VD_Y \rightarrow VD_X$ given by $f(v') = v$ if there is a tuple $t' \in T'$ such that $t'[X] = v$, $t'[Y] = v'$ is well defined and is an injection. \square

In the next proposition, we show the bidirectional implication of the spFD in the singular attributes directed graph of spFD's.

Proposition 11. *Let T be an instance over the relation R . If the spFD's between singular attributes $X_i \in R$, for $i = 1, 2 \dots, w$, form a directed circle in the spFD graph, that is $T \models X_i \rightarrow_{sp} X_{i+1}$ and $T \models X_w \rightarrow_{sp} X_1$. Then the reverse direction of the spFD circle also holds in T , i.e. $T \models X_{i+1} \rightarrow_{sp} X_i$ and $T \models X_1 \rightarrow_{sp} X_w$.*

Proof. As the spFD's form a circle in the spFD graph, then all the attributes have the same number of values in their visible domains. Indeed, by $T \models X_i \rightarrow_{sp} X_{i+1}$, we have $|VD_{X_i}| \geq |VD_{X_{i+1}}|$ for any i , and by $T \models X_w \rightarrow_{sp} X_1$, we have $|VD_{X_w}| \geq |VD_{X_1}|$. So, $|VD_{X_i}| = |VD_{X_{i+1}}|$ for any i . Hence, by Proposition 9, the other direction for each spFD is also satisfied by T . \square

The bidirectional implication of the spFD between the singular attributes does not show that the complements of their weak similarity graphs are the same, For example, $Y \rightarrow_{sp} Z$ and $Z \rightarrow_{sp} Y$ hold in the table below, but $\overline{G_Y} \neq \overline{G_Z}$. On the

Table 7: Possible types of tuples

X	Y
1	a_1
1	\perp
2	a_2
2	\perp
\vdots	\vdots
k	a_k
k	\perp
$k+1$	\perp
\vdots	\vdots
ℓ	\perp
\perp	a_1
\vdots	\vdots
\perp	a_k
\perp	b_1
\vdots	\vdots
\perp	b_r

	X	Y
t_1	1	\perp
t_2	\perp	a_2
t_3	1	a_1
t_4	2	\perp
t_5	\perp	a_2
t_6	\perp	\perp

other hand, $X \rightarrow_{sp} Y$ and $X \rightarrow_{sp} Z$ hold, but $X \not\rightarrow_{sp} YZ$.

X	Y	Z
1	1	\perp
\perp	2	1
2	\perp	2

Propositions 9 and 11 allow us to introduce the following rule.

Digraph rule Let $G = (R, E)$ be a directed graph. If $(A_i, A_j) \in E$ and A_i and A_j are in the same strongly connected component of G , then $(A_j, A_i) \in E$ holds, as well.

Theorem 6. Let T be an SQL table over scheme $R = \{A_1, A_2, \dots, A_n\}$. If $G = (R, E)$ is a directed graph defined by $(A_i, A_j) \in E \iff T \models \{A_i\} \rightarrow_{sp} \{A_j\}$ for attributes $A_i, A_j \in R$, then $G = (R, E)$ satisfies the Digraph rule. Furthermore, for every directed graph $G = (R, E)$ that satisfies the Digraph rule there exists an SQL table T such that $(A_i, A_j) \in E \iff T \models \{A_i\} \rightarrow_{sp} \{A_j\}$.

The proof of Theorem 6 is based on a series of propositions.

Proposition 12. Let T be an SQL table over scheme $R = \{A_1, A_2, \dots, A_n\}$ and let $G = (R, E)$ be the directed graph defined by $(A_i, A_j) \in E \iff T \models \{A_i\} \rightarrow_{sp} \{A_j\}$ for attributes $A_i, A_j \in R$. Then $G = (R, E)$ satisfies the Digraph rule.

Proof. Let $(A_i, A_j) \in E$ be an edge of the directed graph defined by spFD's between singular attributes so that A_i and A_j are in the same strongly connected component of G . This means that there exists a directed path $A_j = X_1, X_2, \dots, X_w = A_i$ in G , thus together with edge (A_i, A_j) a directed cycle is obtained. So by Proposition 11, the other direction of the spFD's hold, i.e $T \models A_j \rightarrow_{sp} A_i$. \square

Proposition 13. *Let $G = (R, E)$ be a strongly connected graph that satisfies the Digraph rule. Then there exists an SQL table T over schema $R = \{A_1, A_2, \dots, A_n\}$ such that $(A_i, A_j) \in E \iff T \models \{A_i\} \rightarrow_{sp} \{A_j\}$. Furthermore, we may assume that $VD(A_i) = \{1, 2\}$ for all $i = 1, 2, \dots, n$ in T and that if $(A_i, A_j) \notin E$, then there exists two rows of T that agree and non-null in A_i and differ and non-null in A_j .*

Proof. Note that a strongly connected graph $G = (R, E)$ satisfies the Digraph rule iff $(A_i, A_j) \in E \iff (A_j, A_i) \in E$. In particular, for any induced subgraph $G' = (R', E')$ of $G = (R, E)$ also satisfies the Digraph rule. We may assume without loss of generality that $(A_1, A_2) \in E$. Let $G_k = (R_k, E_k)$ be the subgraph of $G = (R, E)$ induced by $R_k = \{A_1, A_2, \dots, A_k\}$. We use induction on k to prove that there exists an SQL table T_k over R_k such that $(A_i, A_j) \in E_k \iff T_k \models \{A_i\} \rightarrow_{sp} \{A_j\}$ and if $(A_i, A_j) \notin E$, then there exists two rows of T_k that agree and non-null in A_i and differ and non-null in A_j . The base case $k = 2$ is trivial

$$T_2 = \begin{array}{|c|c|} \hline A_1 & A_2 \\ \hline 1 & 1 \\ \hline 2 & \perp \\ \hline \perp & 2 \\ \hline \end{array}$$

so $T_2 = \{t_0, t_1, t_2\}$ with $t_0[A_j] = 1$ for $j \in \{1, 2\}$ and

$$t_i[A_j] = \begin{cases} 2 & \text{if } i = j \\ \perp & \text{if } i \neq j \end{cases}.$$

Note that T_2 has one entry 2 in each column, and these entries are in different rows, in column A_i the 2 is in row t_i . Now assume that $T_k = \{t_0, t_1, t_2, \dots, t_k\}$ exists for $G_k = (R_k, E_k)$ such that column A_i has its only entry 2 in row t_i and consider G_{k+1} . We add new column A_{k+1} and a new row (tuple) t_{k+1} to T_k as follows.

$$t_i[A_{k+1}] = \begin{cases} 1 & \text{if } (A_{k+1}, A_i) \notin E_{k+1} \\ \perp & \text{if } (A_{k+1}, A_i) \in E_{k+1} \end{cases} \text{ for } i = 1, 2, \dots, k \text{ and } t_{k+1}[A_{k+1}] = 2.$$

Furthermore

$$t_{k+1}[A_i] = \begin{cases} 1 & \text{if } (A_i, A_{k+1}) \notin E_{k+1} \\ \perp & \text{if } (A_i, A_{k+1}) \in E_{k+1} \end{cases} \text{ for } i = 1, 2, \dots, k$$

$$T_{k+1} = \begin{array}{|c|c|c|c|c|} \hline A_1 & A_2 & \dots & A_k & A_{k+1} \\ \hline 1 & 1 & 1 \dots 1 & 1 & 1 \\ \hline 2 & \perp & 1 \dots 1 & 1 & 1 \\ \hline \perp & 2 & 1 \dots \perp & 1 & \perp \\ \hline 1 & \perp & 2 \dots 1 & \perp & \perp \\ \hline \vdots & \vdots & \ddots & \vdots & \vdots \\ \hline \perp & 1 & 1 \dots \perp & 2 & 1 \\ \hline 1 & \perp & 1 \dots \perp & 1 & 2 \\ \hline \end{array}$$

This table satisfies the requirements, since no $A_i \rightarrow_{sp} A_j$ for $1 \leq i, j \leq k$ is destroyed by row t_{k+1} by imputing 1's in place of \perp 's in the last row. Similarly, if we impute a 2 in place of \perp in the last column in row t_i and everywhere else 1's in that column, then we get a strongly possible world showing $A_{k+1} \rightarrow_{sp} A_i$. Furthermore imputing 2 in place of \perp 's in t_{k+1} we get $A_i \rightarrow_{sp} A_{k+1}$ exactly if $(A_i, A_{k+1}) \in E_{k+1}$. \square

Proof. (of Theorem 6). We use induction on the number t of strongly connected components of $G = (R, E)$. The base case $t = 1$ is the statement of Proposition 13. Let the strongly connected components of $G = (R, E)$ be C_1, C_2, \dots, C_t , we may assume without loss of generality, that they are in a topological sort order, that is if $(A_i, A_j) \in E$ and $A_i \in C_i$ and $A_j \in C_j$, then $i \leq j$. There exists an SQL table T_i for each C_i such that $\{A, B\} \in E(C_i) \iff T_i \models A \rightarrow_{sp} B$. Furthermore, by Proposition 13 we have that entries in T_i are 1, 2, \perp , if $A \rightarrow_{sp} B$, 1's in column A match 1's or \perp in B , and 2's are not in the same row with a 1 in these 2 columns. If $A \not\rightarrow_{sp} B$, then there exist two rows as following:

$$\begin{array}{|c|c|} \hline A & B \\ \hline 1 & 1 \\ \hline 1 & 2 \\ \hline \end{array}$$

Assume by induction, we have a table T^{t-1} for the subgraph induced by $C_1 \cup C_2 \cup \dots \cup C_{t-1}$. Construct T^t as follows. First position T^{t-1} and T_t on disjoint row and column sets so that T^{t-1} is extended by 1's in the columns of C_t . In the rows of T_t put matrix $M(r_1, r_2, \dots, r_k)$, which has entry r_i in its i^{th} row for each column of $C_1 \cup C_2 \cup \dots \cup C_{t-1}$, where r_1, r_2, \dots, r_k are pairwise distinct numbers, different from anything appearing in T^{t-1} :

T^{t-1}	1
$M(r_1, r_2, \dots, r_k)$	T_t

Then we add two rows, \mathbf{t}_1^A and \mathbf{t}_2^A for each $A \in C_1 \cup C_2 \cup \dots \cup C_{t-1}$ as follows, $\forall B \in C_1 \cup C_2 \cup \dots \cup C_{t-1}$ and for all $X, Y \in C_t$ if $A \rightarrow_{sp} X$ and $A \not\rightarrow_{sp} Y$.

	A	B	X	Y
\mathbf{t}_1^A	r_A	\perp	\perp	1
\mathbf{t}_2^A	r_A	\perp	\perp	2

where r_A is a new value not appearing in T^{t-1} , also $r_A \neq r_i \forall i = 1, 2, \dots, k$ and if $A \neq B$ then $r_A \neq r_B$.

Let T^t be the table obtained. We check pairs of attributes A, X that $T^t \models A \rightarrow_{sp} X \iff (A, X) \in E$.

Case 1: Both $A, X \in C_1 \cup C_2 \cup \dots \cup C_{t-1}$. If $(A, X) \notin E$, then by the induction hypothesis there exist two rows of T^{t-1} that are both not null in both A and X and agree in A and differ in X , so $T^t \not\models A \rightarrow_{sp} X$. On the other hand, if $(A, X) \in E$, then the induction hypothesis provides an spWorld T^* of T^{t-1} that $T^* \models A \rightarrow X$. Columns A and X can be extended identically below T^{t-1} so that values on those rows in A and X differ from values in T^* , thus imputing any values in place of nulls in the other columns we get an spWord T' of T^t such that $T' \models A \rightarrow X$.

Case 2: Both $A, X \in C_t$. If $(A, X) \notin E$, then by Proposition 13 there are two rows of T_t that they are both 1 in A and take different values (1 and 2) in X , so $T^t \not\models A \rightarrow_{sp} X$. On the other hand, if $(A, X) \in E$, then there is an spWord of T_t that has columns A and X identical. The rows above T_t are also identical in A and X and these two columns can be extended identically below T_t , as well so that 1's in A match 1's in X and the same holds for entries 2. Then other null values of T^t can be arbitrarily substituted from visible domains of the given attributes and an spWord is obtained where $A \rightarrow X$ holds, that is $T^t \models A \rightarrow_{sp} X$.

Case 3: $A \in C_1 \cup C_2 \cup \dots \cup C_{t-1}$ and $X \in C_t$. In this case $(X, A) \notin E$ holds, and there are two rows in T^{t-1} that have different not null values in A , but these two rows take value 1 in X , so $T^t \not\models X \rightarrow_{sp} A$. If $(A, X) \notin E$, then rows \mathbf{t}_1^A and \mathbf{t}_2^A have identical values in A but different values in X , so $T^t \not\models A \rightarrow_{sp} X$. On the other hand, if $(A, X) \in E$, then the nulls of X in rows \mathbf{t}_1^A and \mathbf{t}_2^A can be substituted by 1's, similarly for any rows \mathbf{t}_1^B and \mathbf{t}_2^B that contain nulls in X . If for some $B \in C_1 \cup C_2 \cup \dots \cup C_{t-1}$ the corresponding rows \mathbf{t}_1^B and \mathbf{t}_2^B contain 1 and 2 in column X (i.e. $(B, X) \notin E$), then in \mathbf{t}_1^B we substitute 1 in column A ,

and in \mathbf{t}_2^B substitute r_i in place of the null in column A , where r_i is the value in $M(r_1, r_2, \dots, r_k)$ that stands in the unique row of T_t which has a 2 in column X . The rest of the nulls of T^t can be imputed arbitrarily from the appropriate visible domains, the spWord of T_t obtained satisfies functional dependency $A \rightarrow X$. \square

6 Conclusions

Entering incomplete tuples are allowed into many recent systems' databases. Several research work studied the keys and functional dependencies constraints over incomplete data, such as p/c-keys [22], spKeys [3], c-FDs [23], p-FDs[26], and s/w-FDs [25].

This paper continued the research work started in [3] of strongly possible worlds of SQL tables with NULL's. We introduced a polynomial-time algorithm that checks whether a given set K of attributes is a spKey or not. We also proved that it is NP-complete to do the same for an arbitrary system of $\Sigma = \{sp\langle K_i \rangle : i = 1, 2, \dots, n\}$ of spKey constraints. On the other hand, we showed that Σ can be verified in polynomial time if the sets K_i are pairwise disjoint. Further, we studied strongly possible functional dependencies that were introduced in [4] earlier, as a functional dependency constraint in an SQL table containing NULL's, using the visible domain concept. A graph-theoretical characterization using list coloring is given that can be employed to check when a given spFD holds. This characterization allowed us to prove that verifying whether a single spFD $X \rightarrow_{sp} Y$ holds is NP-complete. This is in sharp contrast with spKey problem, where a single key can be checked in polynomial time.

In another paper [5] spFD properties and axioms analogous to those of weak and strong FD's given by Levene et.al. are introduced and suitable weakenings were given for those axioms that are non sound for spFD's. Here we summarize those in a table for completeness and extend the investigation on sp-Keys and sp/c-FDs together to obtain weakening and transitivity interaction rules.

Our study shows that the spFD's between singular attributes have special properties, because the visible domain represents all the possible extensions for any NULL occurrence. There is a natural correspondence between directed graphs and single attribute spFD's, a characterization of those directed graphs that may occur in this context was given.

The properties of spFD's listed form a step toward a possible axiomatization of spFD's. The main challenge of applying the visible domain is that different spFD's in the premises of rules may hold in different and incompatible strongly possible worlds. More investigation is needed to find how to incorporate this into the axiom system. A first step would be to resolve the following open problem.

Question 1. *Let us assume that $R_S = \emptyset$. Do Reflexivity, Augmentation, Decomposition, Disjoint composition, and Digraph rule form a complete system of inference rules for strongly possible functional dependencies in this case?*

A further simplification that could be attacked using Theorem 3 is when we

assume that there exist no spFD's between singleton attributes. Our experience shows that in that case spFD's are mostly independent of each other, so the first four rules of the previous question could be a complete set for inferences. Another future research direction is defining the closures concerning spFD's. Where the usual definition of $X^+ = \{A: T \models X \rightarrow_{sp} A\}$ may result in $T \not\models X \rightarrow_{sp} X^+$ for spFD's, as the union rule is not sound.

Finally, for database tables, lossless decomposition is an important application of FD's to eliminate redundancy and the possibilities of inconsistent updates. Köhler and Link[23] and Wei and Link [28] show how to use c-FD's or embedded FD's for that. Our future work will include a similar investigation for spFD's.

References

- [1] Achs, Ágnes and Kiss, Attila. Fuzzy extension of datalog. *Acta Cybernetica*, 12(2):153–166, 1995. URL: <https://cyber.bibl.u-szeged.hu/index.php/actcybern/article/view/3454>.
- [2] Acuna, Edgar and Rodriguez, Caroline. The treatment of missing values and its effect on classifier accuracy. In *Classification, clustering, and data mining applications*, pages 639–647. Springer, 2004. DOI: 10.1007/978-3-642-17103-1_60.
- [3] Alattar, Munqath and Sali, Attila. Keys in relational databases with nulls and bounded domains. In *European Conference on Advances in Databases and Information Systems*, pages 33–50. Springer, 2019. DOI: 10.1007/978-3-030-28730-6_3.
- [4] Alattar, Munqath and Sali, Attila. Functional dependencies in incomplete databases with limited domains. In *International Symposium on Foundations of Information and Knowledge Systems*, pages 1–21. Springer, 2020. DOI: 10.1007/978-3-030-39951-1_1.
- [5] Alattar, Munqath and Sali, Attila. Towards an axiomatization of strongly possiblefunctional dependencies. *The Vietnam Journal of Computer Science*, 8(1):133–151, 2021. DOI: 10.1142/S2196888821500056.
- [6] Atzeni, Paolo and Morfuni, Nicola M. Functional dependencies and constraints on null values in database relations. *Information and Control*, 70(1):1–31, 1986. DOI: 10.1016/S0019-9958(86)80022-5.
- [7] Chang, Gang and Ge, Tongmin. Comparison of missing data imputation methods for traffic flow. In *Proceedings 2011 International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE)*, pages 639–642. IEEE, 2011. DOI: 10.1109/TMEE.2011.6199284.
- [8] Cheng, Ching-Hsue, Wei, Liang-Ying, and Lin, Tzu-Cheng. Improving relational database quality based on adaptive learning method for estimating

- null value. In *Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, pages 81–81. IEEE, 2007. DOI: 10.1109/ICICIC.2007.350.
- [9] Codd, Edgar F. *The relational model for database management: version 2*. Addison-Wesley Longman Publishing Co., Inc., 1990.
- [10] Date, CJ. Not is not "not"! (notes on three-valued logic and related matters), chapter 8. In *Relational Database Writings*. Addison-Wesley, Reading, MA, 1989.
- [11] Dhevi, AT Sree. Imputing missing values using inverse distance weighted interpolation for time series data. In *2014 Sixth International Conference on Advanced Computing (ICoAC)*, pages 255–259. IEEE, 2014. DOI: 10.1109/ICoAC.2014.7229721.
- [12] Farhangfar, Alireza, Kurgan, Lukasz, and Dy, Jennifer. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, 41(12):3692–3705, 2008. DOI: 10.1016/j.patcog.2008.05.019.
- [13] Farhangfar, Alireza, Kurgan, Lukasz A, and Pedrycz, Witold. Experimental analysis of methods for imputation of missing values in databases. In *Intelligent Computing: Theory and Applications II*, volume 5421, pages 172–182. International Society for Optics and Photonics, 2004. DOI: 10.1117/12.542509.
- [14] Farhangfar, Alireza, Kurgan, Lukasz A, and Pedrycz, Witold. A novel framework for imputation of missing values in databases. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(5):692–709, 2007. DOI: 10.1109/TSMCA.2007.902631.
- [15] Grahne, Gösta. Dependency satisfaction in databases with incomplete information. In *Proceedings of the 10th International Conference on Very Large Data Bases*, pages 37–45, 1984.
- [16] Grzymala-Busse, Jerzy W. and Hu, Ming. A comparison of several approaches to missing attribute values in data mining. In *International Conference on Rough Sets and Current Trends in Computing*, pages 378–385. Springer, 2000.
- [17] Hartmann, Sven and Link, Sebastian. The implication problem of data dependencies over SQL table definitions: Axiomatic, algorithmic and logical characterizations. *ACM Trans. Database Syst.*, 37(2), 2012. DOI: 10.1145/2188349.2188355.
- [18] Imielinski, Tomasz. Incomplete information in logical databases. *IEEE Data Engineering Bulletin*, 12(2):29–40, 1989.
- [19] Imielinski, Tomasz and Lipski, Witold. Incomplete information in relational databases. *Journal of the ACM*, 31(4):761–791, 1984. DOI: 10.1145/1634.1886.

- [20] Jansen, Klaus and Scheffler, Petra. Generalized coloring for tree-like graphs. *Discrete Applied Mathematics*, 75(2):135–155, 1997. DOI: 10.1016/S0166-218X(96)00085-6.
- [21] Kiss, A. and Márkus, T. Functional and inclusion dependencies and their implication problems. In *Proceedings of the 10th International Seminar on Database Management Systems*, pages 31–38, 1987.
- [22] Köhler, Henning, Leck, Uwe, Link, Sebastian, and Zhou, Xiaofang. Possible and certain keys for SQL. *The VLDB Journal*, 25(4):571–596, 2016. DOI: 10.1007/s00778-016-0430-9.
- [23] Köhler, Henning and Link, Sebastian. SQL schema design: Foundations, normal forms, and normalization. *Information Systems*, 76:88–113, 2018. DOI: 10.1016/j.is.2018.04.001.
- [24] Kratochvil, Jan and Tuza, Zsolt. Algorithmic complexity of list colorings. *Discrete Applied Mathematics*, 50(3):297–302, 1994. DOI: 10.1016/0166-218X(94)90150-3.
- [25] Levene, Mark and Loizou, George. Axiomatisation of functional dependencies in incomplete relations. *Theoretical Computer Science*, 206(1):283–300, 1998. DOI: 10.1016/S0304-3975(98)80029-7.
- [26] Lien, Y Edmund. On the equivalence of database models. *Journal of the ACM*, 29(2):333–362, 1982. DOI: 10.1145/322307.322311.
- [27] Vassiliou, Yannis. Functional dependencies and incomplete information. In *Proceedings of the 6th VLDB Conference*. Morgan Kaufmann Publishers, 1980.
- [28] Wei, Ziheng and Link, Sebastian. Embedded functional dependencies and data-completeness tailored database design. *Proceedings of the VLDB Endowment*, 12(11):1458–1470, 2019. DOI: 10.14778/3342263.3342626.
- [29] Zhang, Shichao, Qin, Zhenxing, Ling, Charles X, and Sheng, Shengli. "Missing is useful": Missing values in cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1689–1693, 2005. DOI: 10.1109/TKDE.2005.188.

Received 16th August 2020

Verified Integration of Differential Equations with Discrete Delay

Andreas Rauh^a and Ekaterina Auer^b

Abstract

Many dynamic system models in population dynamics, physics and control involve temporally delayed state information in such a way that the evolution of future state trajectories depends not only on the current state as the initial condition but also on some previous state. In technical systems, such phenomena result, for example, from mass transport of incompressible fluids through finitely long pipelines, the transport of combustible material such as coal in power plants via conveyor belts, or information processing delays. Under the assumption of continuous dynamics, the corresponding delays can be treated either as constant and fixed, as uncertain but bounded (fixed or time-varying), or even as state-dependent. In this paper, we restrict the discussion to the first two classes and provide suggestions on how interval-based verified approaches to solving ordinary differential equations can be extended to encompass such delay differential equations. Selected close-to-life examples illustrate the theory from the perspective of robustness analysis in engineering applications.

Keywords: interval analysis, delay differential equations, uncertainty, dynamic systems, verified methods

1 Introduction

Delay differential equations arise in many areas of computational science and engineering. Representative examples can be found in the area of modeling biological processes [3], in the area of transport of incompressible substances [17], or in the area of control engineering if signal processing or communication delays are considered [13]. The latter are especially important in the field of networked control systems, where temporally varying communication delays are omnipresent [32]. Long-distance communications included in a closed-loop control framework belong

^aCarl von Ossietzky Universität Oldenburg, Department of Computing Science, Group: Distributed Control in Interconnected Systems, D-26111 Oldenburg, Germany, E-mail: Andreas.Rauh@uni-oldenburg.de, ORCID: 0000-0002-1548-6547

^bUniversity of Technology, Business and Design, Department of Electrical Engineering, D-23966 Wismar, Germany, E-mail: Ekaterina.Auer@hs-wismar.de, ORCID: 0000-0003-4059-3982

to the same class of system models, with the most spectacular applications to be found in the fields of tele-operation [1] in medical surgery (i.e., communication and haptic device feedback between various places or even continents) or in space exploration where human operators take part as generators for reference signals or as decision makers.

A common example of the use of delay differential equations is the representation of the population dynamics in which species growth and mortality rates depend not only on the current state values but also on state information that is delayed by a certain finite time span. This helps to take into account the fact that each species first has to reach fertility age before it takes part in the reproduction process [5,6,9]. As mentioned before, similar considerations appear not only in the mathematical modeling of biological reproduction processes but also in epidemiological models or in (technical) control systems. In control, delay equations are employed when control actions depend additionally on certain previous state information due to communication delays, non-negligible time spans for information acquisition and processing, transport phenomena of physical substances or when control actions are decided upon based on (averaged) previous state information.

In this paper, we restrict the discussion to the case in which the system model has a single, finitely long, discrete delay. Then, the dynamic system model can be stated as

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{x}(t - \tau^*)) , \quad \mathbf{x}(t) \in \mathbb{R}^n , \quad \tau^* \geq 0 \quad (1)$$

with

$$\mathbf{f} : \mathbb{R} \times \mathbb{R}^{2n} \mapsto \mathbb{R}^n . \quad (2)$$

Aside from the initial condition

$$\mathbf{x}(0) = \mathbf{x}_0 \quad (3)$$

at the single point $t = 0$, knowledge about a state initialization function

$$\mathbf{x}^*(t) := \mathbf{x}(t) \quad (4)$$

for the time span $-\tau^* \leq t \leq 0$ is required to determine a unique solution. Throughout this paper, we assume that the initialization function evaluated for $t = 0$ provides the same value as the point-valued initial condition \mathbf{x}_0 , leading to a solution $\mathbf{x}(t)$ that is continuous at $t = 0$. Moreover, the $*$ symbol consistently denotes exact solutions to the simulation models under consideration and precisely known values for the delay.

A typical floating-point solution procedure for such system models is the so-called *method of steps* [33] in which the problem (1) is transformed into the non-autonomous initial value problem

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{x}^*(t - \tau^*)) , \quad \text{with } \mathbf{x}(0) = \mathbf{x}^*(0) , \quad (5)$$

that is used to predict the temporal evolution of $\mathbf{x}(t)$ over the time interval $t \in [0; \tau^*]$ for the already known delayed state information $\mathbf{x}^*(t - \tau^*)$. For successive

intervals of the length τ^* , the procedure can be continued by utilizing the result from the previous time slice as the new initialization function.

This procedure for a floating point-based approximation of the solution to the delay differential equation is restricted to the case in which the initial state is given by a specific point in the state space and the delay time is known exactly. Uncertainty in the initial system states (or in the initialization function, respectively) could be treated by repeated simulations in a Monte-Carlo-like manner. Here, a well-known disadvantage is a potentially large computational effort that does not allow for determining verified outer bounds for the sets of reachable states if interval bounds for the initial conditions \mathbf{x}_0 as well as for the initialization function $\mathbf{x}^*(t)$ over the time interval $t \in [-\tau^*; 0]$ are given [20]. The same also holds for uncertain, but bounded delays τ^* which are themselves given as intervals. Two practice-relevant cases need to be distinguished here:

1. The delay is uncertain but constant over each time slice of a solution approach corresponding to the above-mentioned method of steps.
2. The delay is bounded from below and above, but may vary arbitrarily within these bounds.

Note that the second case is also strongly linked to scenarios in which only bounds for the initialization function (4) are available but the exact temporal evolution in the past is unknown.

From both a methodological and practical point of view, it is crucial to investigate such phenomena because increasing the delay time (for example, in the feedback path of a closed-loop control system) may turn a system with aperiodic dynamics into a system with oscillatory behavior. In addition, the introduction of delay may also turn asymptotically stable systems into unstable ones. The stability investigation of systems with delay, however, is not trivial and still a subject for ongoing research. For possible references on this topic, see [7, 9, 18, 21].

The main contribution of this paper is the generalization of verified solution techniques for classical, delay-free sets of ordinary differential equations to both cases mentioned above, namely, systems with constant as well as temporally varying but bounded delays. The general solution approach is derived exemplarily for an exponential state enclosure technique published by the authors in [29]. This approach makes use of techniques from the field of interval analysis [11, 19] to compute outer bounds that rigorously enclose all possible state trajectories of uncertain dynamic system models.

In contrast to the existing techniques with result verification for solving delay differential equations [10, 35, 36] (that employ Taylor methods or radii polynomial approaches), we do not focus on obtaining especially tight enclosures, which is necessary for a computational proof of such properties as periodicity of solutions. Instead, we aim at computing guaranteed outer solution enclosures by an approach that represents state trajectories by simple (exponential) functions in a computationally cheap way. It should be pointed out that the reduced complexity (resulting from the simple exponential state enclosures) would allow for an easier

reimplementation of the whole algorithm on the GPU [2]¹ or for development of online-adjustable control strategies in the frame of model-predictive control. Additionally, our method is directly applicable to scenarios in which parameters and the delay can vary temporally but stay bounded.

This paper is structured as follows. In Sec. 2, there is an overview of the initial value problems for delay differential equations to be solved by the approach suggested in this paper. Before detailed solution procedures are presented in Secs. 4 and 5, the interval-based exponential enclosure technique for classical ordinary differential equations from [29] is summarized in Sec. 3. This is a representative solution approach which is extended in this paper to the case of systems with a finite delay time. In Sec. 6, various numerical examples are presented including systems with exactly known delay times, with an uncertain but constant delay, and with a time-varying bounded delay. Finally, conclusions and an outlook on future work are given in Sec. 7.

2 Problem Formulation

Throughout this paper, the following variants of the delay differential equation model (1)–(4) are considered.

DDE1 The initial conditions and the initialization function in (3), (4), respectively, are both uncertain but bounded. The initial states are assumed to be included in the interval²

$$\mathbf{x}_0 \in [\underline{\mathbf{x}}_0 ; \overline{\mathbf{x}}_0] \quad , \quad (6)$$

where the component-wise defined inequalities $\underline{x}_{0,i} \leq \overline{x}_{0,i}$, $i \in \{1, \dots, n\}$, hold. Analogously, the initialization function (4) is supposed to be given by the bounds³

$$\mathbf{x}^*(t) \in [\mathbf{x}_0] \quad \text{with} \quad \frac{d\mathbf{x}^*(t)}{dt} = \mathbf{0} \quad \text{for all} \quad t \in [-\tau^* ; 0] \quad . \quad (7)$$

In contrast to considering interval bounds for the initial states and for the initialization function, it is assumed that the delay $\tau^* > 0$ is precisely known⁴.

¹Such GPU implementations, accounting for data parallelism, are especially helpful to solve the task of an experiments-based parameter identification of dynamic systems.

²If necessary, we use the compact notation $[\mathbf{x}_0]$ throughout this paper to abbreviate the interval $[\underline{\mathbf{x}}_0 ; \overline{\mathbf{x}}_0]$. Bold face characters are employed to distinguish vectors (lower case) and matrices (upper case) from scalar variables.

³The algorithms presented in the following are not restricted to temporally constant initialization functions. They are chosen in this paper mainly to simplify the notation. Non-constant initializations arise naturally at each time instant $t > 0$ at which the integration is restarted if step size control strategies or multi-step simulations are performed.

⁴In **DDE1**, it is assumed that the delay time can be represented exactly by a machine number in the software implementation of the solution approach. If this is not the case, the formulation from **DDE2** can be used instead, where the point value τ^* is enclosed in a tight interval of machine numbers.

DDE2 As in **DDE1**, the initial system states \mathbf{x}_0 are assumed to be given by (6). However, the delay is now considered to be uncertain but temporally constant according to

$$\tau^* \in [\underline{\tau}^* ; \bar{\tau}^*] \quad \text{with} \quad \underline{\tau}^* < \bar{\tau}^* . \tag{8}$$

Therefore, the initialization function of **DDE1**, cf. (7), needs to be adapted in such a way that interval bounds are available for all $t \in [-\bar{\tau}^* ; 0]$. A special case of this definition arises for a delay-free lower bound $\underline{\tau}^* = 0$.

DDE3 This scenario is almost identical to **DDE2** except that the delays are not temporally constant. Now, the delay

$$\tau^* \in [\underline{\tau}^* ; \bar{\tau}^*] \quad \text{with} \quad 0 \leq \underline{\tau}^* < \bar{\tau}^* \tag{9}$$

may vary arbitrarily between its lower and upper bounds. No information on the temporal variation rate is available in this setting. Obviously, this is also true for the state initialization function, where arbitrary variations of \mathbf{x}^* have to be accounted for within the respective interval bounds.

3 Verified Simulation Routine for Asymptotically Stable Delay-Free Ordinary Differential Equations

Delay-free state equations can often be assumed to be asymptotically stable in control engineering applications since, if they are not, an appropriate (state) feedback control law can be designed, in many cases with quasi-linear state-space representations. For finding enclosures of the solutions to such problems, the authors developed a verified exponential state enclosure technique [29] summarized briefly in this section. In the following sections, we extend this example of a solution procedure to the case of delay differential equations since they also play an important role in the area of control. However, any other verified approach for solving initial value problems for ordinary differential equations (e.g., from [14, 15, 20]) can be generalized analogously for the case of delay differential equations if the strategies described in Secs. 4 and 5 are employed.

Definition 1 (Quasi-linear autonomous model). *For a nonlinear system model*

$$\dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t)) , \tag{10}$$

a quasi-linear dynamic system representation is given by the state-space representation

$$\dot{\mathbf{x}}(t) = \mathbf{A}(\mathbf{x}(t)) \cdot \mathbf{x}(t) , \tag{11}$$

where the reformulation from (10) to (11) is obtained by exactly factoring out the state vector $\mathbf{x}(t)$ from the nonlinear right-hand side $\mathbf{a}(\mathbf{x}(t))$ so that all entries of

$\mathbf{A}(\mathbf{x}(t))$ are well-defined and finite for all reachable states⁵.

Definition 2 (Exponential state enclosure). *The time-dependent exponential enclosure function (indicated by the index e)*

$$\check{\mathbf{x}}(t) \in [\mathbf{x}_e](t) := \exp([\mathbf{\Lambda}] \cdot t) \cdot [\mathbf{x}_e](0), \quad [\mathbf{x}_e](0) = [\mathbf{x}_0] \quad (12)$$

with the parameter matrix

$$[\mathbf{\Lambda}] := \text{diag} \{[\lambda_i]\}, \quad i \in \{1, \dots, n\}, \quad (13)$$

defines a verified exponential state enclosure for the system model (11) with $\mathbf{x}(0) \in [\mathbf{x}_0]$ if it is determined according to Theorem 1. It is then guaranteed to enclose all possible exact state trajectories $\check{\mathbf{x}}(t)$.

Theorem 1 ([25, 29] Iteration for exponential state enclosures). *The exponential state enclosure (12) is guaranteed to contain the set of all reachable states $\check{\mathbf{x}}(T)$ at the point of time $t = T > 0$ according to*

$$\check{\mathbf{x}}(T) \in [\mathbf{x}_e](T) := \exp([\mathbf{\Lambda}] \cdot T) \cdot [\mathbf{x}_e](0), \quad (14)$$

if the elements on the main diagonal of $[\mathbf{\Lambda}]$ are computed by the converging iteration

$$[\lambda_i]^{(\kappa+1)} := \frac{a_i \left(\exp([\mathbf{\Lambda}]^{(\kappa)} \cdot [t]) \cdot [\mathbf{x}_e](0) \right)}{\exp([\lambda_i]^{(\kappa)} \cdot [t]) \cdot [x_{e,i}](0)}, \quad \kappa \in \{0, 1, 2, \dots\}, \quad (15)$$

$i \in \{1, \dots, n\}$, with the prediction horizon $[t] = [0; T]$.

Remark 1. As discussed in [25, 29], the iteration (15) is based on the application of the Picard iteration if the type of solution representations is restricted to the exponential expressions according to Def. 2. Therefore, the system models under consideration need to satisfy the same requirements that are needed for applying a Picard iteration (i.e., applicability of Banach's fixed point theorem) for finding verified state enclosures. Especially, we suppose differentiability of $\mathbf{a}(\mathbf{x})$ on the intervals $\exp([\mathbf{\Lambda}]^{(\kappa)} \cdot [t]) \cdot [\mathbf{x}_e](0)$.

Remark 2. A typical initialization of the iteration (15) is $[\mathbf{\Lambda}]^{(0)} = \text{diag} \{[\lambda_i]^{(0)}\}$, $i \in \{1, \dots, n\}$, where these intervals are centered around the eigenvalues of a linearization of the nonlinear state equations for a representative point from the state enclosure at $t = 0$.

Remark 3. For sufficiently smooth system models (10), this approach can be easily extended to include a differential sensitivity analysis with respect to initial conditions and time-invariant parameters (see [22] for details).

⁵Such quasi-linear reformulations are typically not defined uniquely. For example, there exist infinitely many factorizations of the product $x_1 \cdot x_2 = a_1(x_2) \cdot x_1 + a_2(x_1) \cdot x_2$, where $a_1(x_2) = px_2$ and $a_2(x_1) = (1-p) \cdot x_1$ with $p \in \mathbb{R}$.

Corollary 1 ([25, 29]). *If quasi-linear state-space representations according to Def. 1 are considered, the component-wise reformulation*

$$\dot{x}_i(t) = a_i(\mathbf{x}(t)) = \sum_{j=1}^n a_{ij}(\mathbf{x}(t)) \cdot x_j(t) \tag{16}$$

can be used for the state equations. In this case, the interval-related dependency problem [11], the wrapping effect [16], and the computational effort while using formula (15) can be reduced if it is reformulated symbolically into

$$\begin{aligned} [\lambda_i]^{(\kappa+1)} &:= a_{ii}([\mathbf{x}_e]^{(\kappa)}([t])) \\ &+ \sum_{\substack{j=1 \\ j \neq i}}^n \left\{ a_{ij}([\mathbf{x}_e]^{(\kappa)}([t])) \cdot e^{([\lambda_j]^{(\kappa)} - [\lambda_i]^{(\kappa)}) \cdot [t]} \cdot \frac{[x_{e,j}](0)}{[x_{e,i}](0)} \right\} . \end{aligned} \tag{17}$$

This reformulation is especially beneficial if the system matrix $\mathbf{A}(\mathbf{x}(t))$ is diagonally dominant.

Remark 4. The exponential enclosure technique according to Theorem 1 is applicable to scenarios characterized by solution sets in which the value zero is not contained in the domain of reachable states. This becomes obvious in Eq. (15), where the guaranteed state enclosure appears in the denominator of the iteration formula [25, 28]. For linear systems with oscillatory dynamics and precisely known parameters, this issue can be resolved by a suitable time-invariant change of coordinates as shown in the last example from Sec. 6. For this purpose, Theorem 1 needs to be generalized using complex-valued interval techniques [28, 29]. Alternatively, if the value zero is only *crossed* a finite number of times, enclosures can be obtained by switching to the Picard iteration based VALENCIA-IVP technique or to a low-order Taylor series expansion over the respective time interval.

4 Delay Differential Equations with a Constant, Precisely Known Delay

In general, delay differential equations with bounded uncertainty can be treated by adapting the method of steps to rely on verified solvers for ordinary differential equations instead of floating-point ones. For the case of **DDE1**, this method can be employed directly after defining the time intervals

$$[\mathcal{T}]_m = [m\tau^* ; (m + 1)\tau^*] , \quad m \in \mathbb{N}_0 , \tag{18}$$

with a length that is equal to the a-priori known constant delay τ^* . Then, an exponential state enclosure can be introduced according to Def. 3 for each $[\mathcal{T}]_m$.

Definition 3 (Exponential state enclosure for delay differential equations). *The time-dependent exponential enclosure function for the m -th time interval $t \in [\mathcal{T}]_m$*

$$\check{\mathbf{x}}(t) \in [\mathbf{x}_e]_m(t) := \exp([\mathbf{\Lambda}]_m \cdot (t - m\tau^*)) \cdot [\mathbf{x}_e](m\tau^*) \tag{19}$$

with $[\mathbf{x}_e](0) = [\mathbf{x}_0]$ and the parameter matrix

$$[\mathbf{\Lambda}]_m := \text{diag} \{[\lambda_i]\}_m, \quad i \in \{1, \dots, n\}, \quad (20)$$

defines a verified exponential state enclosure for the delay differential equation of type **DDE1** if it is determined according to Theorem 2. For compatibility with the initialization function (7), $[\mathbf{\Lambda}]_{-1}$ is set to

$$[\mathbf{\Lambda}]_{-1} := \mathbf{0}, \quad \text{corresponding to } [\mathbf{x}_e]_{-1}(t) \equiv [\mathbf{x}_0]. \quad (21)$$

Theorem 2 (Iteration for delay differential equations of type **DDE1**). *The exponential state enclosure (19) is guaranteed to contain the set of all reachable states $\tilde{\mathbf{x}}(T)$ at the point of time $t = T \in [\mathcal{T}]_m$, that is,*

$$\tilde{\mathbf{x}}(T) \in [\mathbf{x}_e]_m(T) := \exp([\mathbf{\Lambda}]_m \cdot (T - m\tau^*)) \cdot [\mathbf{x}_e](m\tau^*), \quad (22)$$

if the elements on the main diagonal of $[\mathbf{\Lambda}]_m$ are computed using the converging iteration

$$[\lambda_i]_m^{(\kappa+1)} := \frac{f_i\left([t], \exp([\mathbf{\Lambda}]_m^{(\kappa)} \cdot ([t] - m\tau^*)) \cdot [\mathbf{x}_e](m\tau^*), [\mathbf{x}_e]_{m-1}([t] - \tau^*)\right)}{\exp([\lambda_i]_m^{(\kappa)} \cdot ([t] - m\tau^*)) \cdot [x_{e,i}](m\tau^*)}, \quad (23)$$

$i \in \{1, \dots, n\}$, with the prediction horizon $[t] = [m\tau^*; T] \subseteq [\mathcal{T}]_m$, where all f_i are defined according to Eq. (1).

Proof. Due to the restriction $T \in [\mathcal{T}]_m$, the third argument $[\mathbf{x}_e]_{m-1}([t] - \tau^*)$ of f_i in (23) depends only on solution enclosures from the previous time interval $[\mathcal{T}]_{m-1}$ and is, therefore, completely known. A non-autonomous initial value problem can be solved at this time step following the general idea of the method of steps (5). Hence, the proof is the same as for Theorem 1 (originally published in [25, 29]), if the iteration (23) is substituted for (15). \square

Remark 5. Overestimation in the elements $[\lambda_i]_m^{(\kappa+1)}$ in (23) appearing due to multiple dependencies on common interval variables can be reduced by exploiting the quasi-linear structure of the problem and reformulating the iteration symbolically (cf. Corollary 1). Another overestimation reduction possibility is to employ a classical subdivision strategy for range computation of interval expressions, also used in global optimization [4]. For that, it is necessary to subdivide the time interval $[t]$ into multiple subintervals, carry out the procedure and, finally, determine the convex interval hull of all resulting enclosures over the subintervals. An additional advantage of subdivision strategies for delay equations is that the time subintervals can also be used while determining the bounds for $[\mathbf{x}_e]_{m-1}([t] - \tau^*) = [\mathbf{x}_e]_{m-1}([(m-1)\tau^*; T - \tau^*])$.

The subdivision strategy described in the remark can be used to control the step size. For this purpose, the integration time horizon $[\mathcal{T}]_m$ is split into multiple shorter time slices and solution parameters $[\lambda_i]_{m,i}$ are computed successively for each of the temporal subintervals $[m\tau^*; m\tau^* + \tau_1]$, $[m\tau^* + \tau_1; m\tau^* + \tau_2]$, \dots , $[m\tau^* + \tau_{l-1}; m\tau^* + \tau_l]$, \dots with $0 < \tau_1 < \tau_2 < \dots < \tau_l < \dots < \tau^*$.

5 Delay Differential Equations with Uncertain Delay

In this section, solution procedures for delay differential equations with uncertain delays are presented. Here, we consider two cases of bounded delays: either temporally constant or arbitrarily varying within the respective interval bounds.

5.1 Delay Differential Equations with a Constant, Interval-Bounded Delay

5.1.1 Systems with Strictly Non-Zero Delay

The approach from Sec. 4 can be extended to cover systems with a strictly non-zero time delay in a straightforward way. For that purpose, we introduce the time intervals

$$[\mathcal{T}]_m = [t_m ; t_{m+1}] , \quad m \in \mathbb{N}_0 , \quad t_0 = 0 , \quad (24)$$

where the infima and suprema of $[\mathcal{T}]_m$ denote the temporal discretization mesh with which the (exponential) solution enclosures for the delay differential equation of type **DDE2** are computed. In this subsection, we further assume that $t_{m+1} - t_m \leq \underline{\tau}^*$ holds. This restriction will be removed in the following subsection, where the case of a possibly vanishing time delay is investigated. The following definition is a generalization of Def. 3 which allows us to represent state enclosures covering multiple points t_m of the temporal discretization mesh.

Definition 4 (Generalized exponential state enclosure for DDEs).

A generalized time-dependent exponential enclosure over a time interval $t \in [\mathcal{T}]_a^b = [t_a ; t_b]$, $t_a \leq t_b$, $t_a \geq 0$, is given by

$$[\mathbf{x}_e]([\mathcal{T}]_a^b) = \begin{cases} [\mathbf{x}_e]_{m_a}([\mathcal{T}]_a^b) & \text{if } m_a = m_b , \\ \bigcup_{j=m_a}^{m_b} [\mathbf{x}_e]_j([\mathcal{T}]_a^b \cap [\mathcal{T}]_j) & \text{otherwise .} \end{cases} \quad (25)$$

Here, the indices m_ι , $\iota \in \{a, b\}$, of the corresponding discretization points are determined according to

$$m_\iota = \max_{m \in \mathcal{Z}} \{m\} \quad \text{with} \quad \mathcal{Z} := \{m \mid m \in \mathbb{N}_0 \text{ and } 0 \leq t_{m_a} \leq t_\iota < t_{m_\iota+1}\} . \quad (26)$$

The individual interval enclosure functions in (25) are given by

$$\check{\mathbf{x}}(t) \in [\mathbf{x}_e]_m(t) := \exp([\mathbf{\Lambda}]_m \cdot (t - t_m)) \cdot [\mathbf{x}_e](t_m) \quad \text{for } t \in [\mathcal{T}]_m \quad (27)$$

with the diagonal parameter matrices $[\mathbf{\Lambda}]_m$ computed as in Theorem 3.

In addition, the definition (25) can be extended to the case $t_b \leq 0$ by setting $[\mathbf{x}_e]([t_a ; t_b]) \equiv [\mathbf{x}_0]$; analogously, $[\mathbf{x}_e]([t_a ; t_b]) \equiv [\mathbf{x}_0] \cup [\mathbf{x}_e]([0 ; t_b])$ holds for $t_a < 0$ and $t_b \geq 0$.

Remark 6. All intersections $[\mathcal{T}]_a^b \cap [\mathcal{T}]_m$ of time intervals in Eq. (25) restrict the domains on which the respective functions are evaluated to the domains on which the parameter matrices $[\mathbf{\Lambda}]_m$ and, hence, the corresponding solution tubes are defined.

Theorem 3 (Iteration for delay differential equations of type **DDE2**).

If the discretization mesh is defined such that $t_{m+1} - t_m \leq \tau^*$, the exponential state enclosure (25) is guaranteed to contain the set of all reachable states $\check{\mathbf{x}}(T)$ at the point of time $t = T \in [\mathcal{T}]_m$, that is,

$$\check{\mathbf{x}}(T) \in [\mathbf{x}_e]_m(T) := \exp([\mathbf{\Lambda}]_m \cdot (T - m\tau^*)) \cdot [\mathbf{x}_e](m\tau^*) \quad , \quad (28)$$

if $[\mathbf{\Lambda}]_m$ is computed by the converging iteration

$$[\lambda_i]_m^{(\kappa+1)} := \frac{f_i\left([t] \ , \ \exp([\mathbf{\Lambda}]_m^{(\kappa)} \cdot ([t] - t_m)) \cdot [\mathbf{x}_e](t_m) \ , \ [\mathbf{x}_e]([t] - [\tau^*])\right)}{\exp([\lambda_i]_m^{(\kappa)} \cdot ([t] - t_m)) \cdot [x_{e,i}](t_m)} \quad , \quad (29)$$

$i \in \{1, \dots, n\}$, with the prediction horizon $[t] = [t_m ; T] \subseteq [\mathcal{T}]_m$.

Proof. In the last argument of the numerator term in (29), $[\mathbf{x}_e]([t] - [\tau^*])$ is independent of the parameter matrix $[\mathbf{\Lambda}]_m^{(\kappa)}$ for the current time interval $[\mathcal{T}]_m$. This means that this term can again be interpreted as an external input to a non-autonomous system of ordinary differential equations. Therefore, the proof is identical to the proof of Theorem 2. \square

Remark 7. For the time interval $[t]$, a subdivision strategy can be used in full analogy to Sec. 4. In addition, the interval $[\tau^*]$ may be subdivided into multiple time intervals, followed by determining separate solution enclosures $[\mathbf{x}_e]_m(T)$ for each delay subinterval when evaluating Eq. (29). In the final step, the convex interval hull over all individual solutions can be determined to describe the set of reachable states. This is a direct consequence of the assumption of an uncertain but temporally constant delay.

Remark 8. Under the assumptions of this subsection, the Definitions 3 and 4 become identical.

5.1.2 Systems Involving Zero Delay

If $0 \in [\tau^*]$ or if a discretization mesh with $t_{m+1} - t_m > \tau^*$ is employed, Theorem 3 needs to be adjusted according to the following Corollary 2.

Corollary 2 (Iteration for **DDE2** with potentially zero delay). *In the case of a potentially vanishing delay, the exponential state enclosure (28) contains the set of all reachable states $\check{\mathbf{x}}(T)$ at the point of time $t = T \in [\mathcal{T}]_m$ if $[\mathbf{\Lambda}]_m$ is set to the outcome of the iteration (29), where the last numerator term $[\mathbf{x}_e]([t] - [\tau^*])$ is evaluated according to Def. 4 with $t_a = \inf([t] - [\tau^*])$ and $t_b = \sup([t] - [\tau^*])$ as a function of all parameters $[\lambda_i]_m^{(\kappa)}$.*

Remark 9. Splitting both intervals $[t]$ and $[\tau^*]$ into subintervals as described in the previous subsection remains admissible due to the time invariance of the delay τ^* .

5.2 Delay Differential Equations with Uncertain, Temporally Varying Delays

Both Theorem 3 and Corollary 2 can be applied to the case of time-varying, uncertain, bounded delays. Note, however, that subdivision strategies suggested as a countermeasure against overestimation in Sec. 5.1 need to be handled with more care.

If the intervals $[t]$ and $[\tau^*]$ are subdivided, it is necessary to compute multiple results $[\lambda_i]_m^{(\kappa+1)}$ followed by the convex hull operation unifying them for each subsequent evaluation of the iterations according to Theorem 3 and Corollary 2. This is the only admissible subdivision strategy for reducing the dependency problem in this case. Note that this subdivision approach is equally valid for reduction of overestimation that is caused by wide intervals $[\lambda_i]_m^{(\kappa)}$ resulting from the previous iteration step.

6 Numerical Examples

In this section, representative application scenarios are presented for the proposed interval technique. They are linear scalar (cf. Sec. 6.1) and multi-dimensional system models (cf. Sec. 6.5) with exactly known delay, a nonlinear process model with exactly known and uncertain delay that is inspired by mathematical models from the field of population dynamics (cf. Sec. 6.2 and 6.3) as well as the simulation of Wright’s equation with an uncertain parameter (cf. Sec. 6.4). The specified values for delay times are assumed to be represented by the closest floating point number, where for the cases of an exactly known time delay this value is an integer multiple of the underlying sampling time. Note that not exactly representable delay times can easily be accounted for by the setting in **DDE2**.

6.1 A System Model with an Exact Analytic Solution

As the first application scenario, consider the dynamic system model [30, Chap. 12]

$$\dot{x}(t) = a \cdot x(t - \tau^*) \tag{30}$$

of type **DDE1** with the exactly known, non-zero delay $\tau^* > 0$. If the initialization function for $t \leq 0$ is equal to the constant $x(t) \equiv x_0$ and if the identical initial condition $x(0) = x_0$ is considered, the exact solution $x_m(t)$ can be computed for each time interval $t \in [\mathcal{T}]_m$, $m \in \mathbb{N}_0$, cf. (18), by applying the method of steps in a recursive manner. For that purpose, the system model (30) is reformulated into

$$\int_{x_{m-1}(m\tau^*)}^{x_m(t)} d\chi = a \cdot \int_{m\tau^*}^t x_{m-1}(\eta - \tau^*) d\eta \tag{31}$$

by separating the variables t and x . This leads to the solution representation

$$x_m(t) = x_{m-1}(m\tau^*) + a \cdot \int_{m\tau^*}^t x_{m-1}(\eta - \tau^*) d\eta, \quad (32)$$

where both formulas (31) and (32) are initialized with $x_{-1}(t) \equiv x_0$. Evaluating the expression (32) at integer multiples of the delay time, i.e., for $t = m\tau^*$, $m \in \mathbb{N}_0$, yields the closed-form solution representation

$$x(t) = \begin{cases} x_0 & \text{for } t = 0 \\ x_0 \cdot (a\tau^* + 1) & \text{for } t = \tau^* \\ \frac{x_0}{2} \left((a\tau^*)^2 + 4a\tau^* + 2 \right) & \text{for } t = 2\tau^* \\ \frac{x_0}{6} \left((a\tau^*)^3 + 12(a\tau^*)^2 + 18a\tau^* + 6 \right) & \text{for } t = 3\tau^* \\ \frac{x_0}{24} \left((a\tau^*)^4 + 32(a\tau^*)^3 + 108(a\tau^*)^2 + 96a\tau^* + 24 \right) & \text{for } t = 4\tau^* \\ \dots & \dots \end{cases} \quad (33)$$

The result (33) can be extended by the well-known rules of interval analysis to outer state enclosures, if uncertainty in the initial state $x_0 \in [x_0]$ and in the time-invariant parameter $a \in [a]$ needs to be accounted for. This interval representation (evaluated in the following in a naive way in terms of the natural interval extension [11] in INTLAB [31]) serves as one of the references to which the novel simulation procedure according to Sec. 4 can be compared.

Fig. 1 provides a comparison of the novel iteration approach with an interval-based evaluation of the analytic solution representation according to Eq. (33). The first simulation result in Fig. 1(a) visualizes the influence of the integration step size on the tightness of the solution obtained by the application of Theorem 2. Exemplarily, the constant integration step sizes $\Delta t = \tau^* = 0.1$ (corresponding to the direct generalization of the method of steps) and $\Delta t = \frac{\tau^*}{200} = 5 \cdot 10^{-4}$ are compared. It can be seen that the simulation for the larger step size breaks down after $t = 2.1$ because overestimation leads to solutions that include the value zero in the denominator of the iteration of Theorem 2. To some extent, this can be avoided by reducing the step size. Hence, the other versions of the example are investigated using this reduced step size. Alternatively, it is possible to apply a different enclosure definition (such as the basic state enclosure of VALENCIA-IVP) during those time spans in which the solution crosses zero.

The subplot in Fig. 1(b) shows that a naive interval extension of the analytic solution representation leads to significantly wider interval bounds than the proposed iteration procedure if a is the only uncertain parameter in the model. This is mainly caused by multiple dependencies on the interval parameter $[a]$ in the analytic solution representation. This dependency effect can be reduced by advanced interval evaluation techniques. In Fig. 1, enhanced enclosures are visualized which are computed by subdividing the parameter domain into an equidistant grid with 100 subintervals for each of the parameters. The dependency is less critical if a

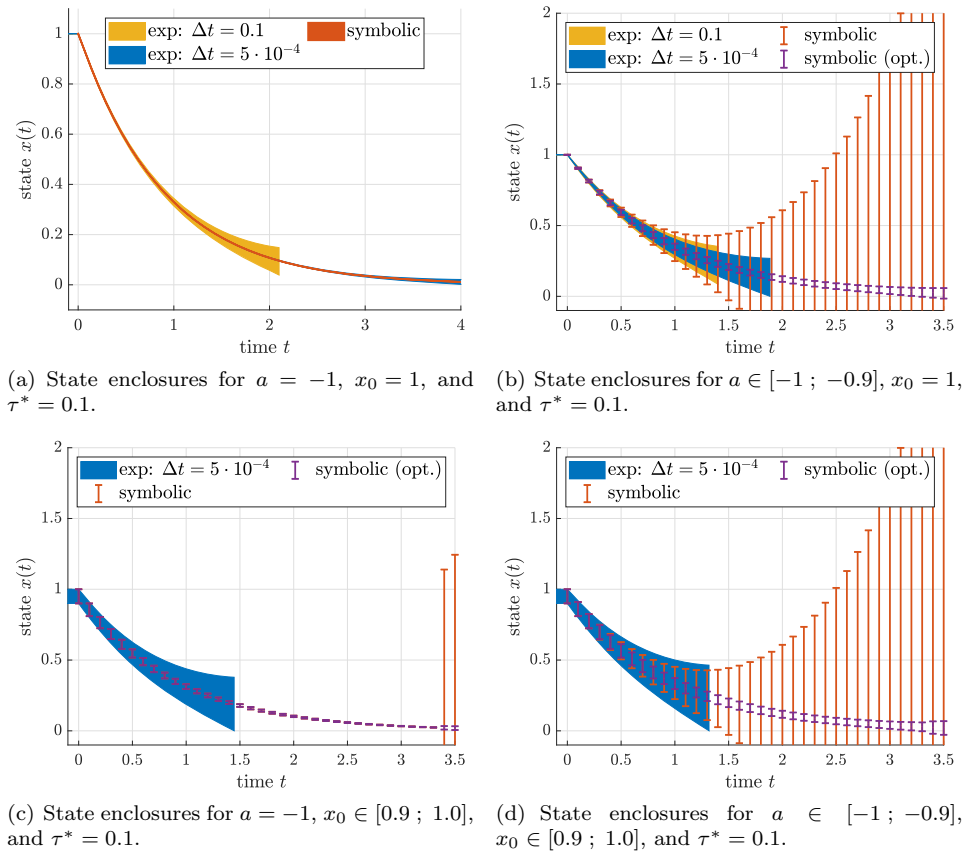


Figure 1: State enclosures for the linear system model (16) with constant delay.

is set to a point value in Fig. 1(c), where the interval evaluation of the analytic solution representation deteriorates rapidly if overestimation leads to the fact that the value zero is included in the solution set. Note that the exponential enclosure technique in this case breaks down before $t = 1.5$. This can be avoided by subdividing the initial state interval and by subsequently performing multiple simulations for the respective subintervals⁶. For the case in Fig. 1(d), where both the initial condition and the system parameter are set to interval quantities, the numerical and analytic solutions are quite similar up to the point where the exponential enclosure technique is no longer valid due to the inclusion of the value zero in the computed state bounds.

⁶As already mentioned in the introduction of this paper, such subdivisions are typically employed anyway for an experiments-based parameter identification of dynamic systems. The proposed solution algorithm can easily be adjusted to a GPU implementation [2] which performs a parallelized simulation for all subintervals by exploiting the concept of data parallelism.

Note that exponential state enclosures computed for a single parameter interval $[a]$ and a single box of initial conditions comprising the bounds on the initialization function $[x_0]$ are generally wider than an optimized interval evaluation of the closed-form solution. This is caused by the fact that Theorem 2, evaluated for a single interval box, provides state enclosures that are valid for arbitrary temporal parameter variations within the respective box, while the analytic solution representation assumes a time invariant parameter with vanishing time derivatives of the initialization function for $t < 0$.

6.2 A Nonlinear System Model with Constant Bounded Delay

As the second application, consider the nonlinear system model

$$\dot{x}(t) = a \cdot x(t) + b \cdot x^3(t - \tau^*) \quad \text{with} \quad x(t) \equiv x_0 \quad \text{for} \quad t \leq 0, \quad (34)$$

where $a \in [a] = [-0.2; -0.1]$, $b \in [b] = [0.01; 0.02]$, $x_0 \in [x_0] = [0.9; 1.0]$, and $\tau^* \in [\tau^*] = [0.1; 1.0]$ are temporally constant interval parameters.

This delay differential equation model reflects a simplified problem from the field of population dynamics [3], where the state variable x represents a species concentration, the parameter a a decay rate (i.e., due to mortality), and b the rate of reproduction depending in a cubic manner on the delayed state information. The parameter τ^* describes the uncertain age of maturation after which the members of the population participate in the reproduction process.

Since the delay parameter is uncertain, the interval-based solution to this model is computed using the methods of Sec. 5.1. A grid-based floating point solution, obtained with the help of the MATLAB routine `dde23` and a maximum step size $\Delta t = 0.01$, is included for comparison in Fig. 2(a). It can be seen that the exponential interval technique encloses the grid-based evaluation in tight lower and upper bounds, where the lower bound especially has almost no overestimation. Note that the grid-based simulation consists of 10^4 individual system simulations because all four uncertain parameters were independently subdivided into 10 points each.

In contrast, the exponential state enclosure was determined without subdividing the interval bounds $[a]$ and $[b]$. For a sake of comparison with the assumption of arbitrary varying initialization functions and delays in the interior of the respective interval bounds (which is the subject of the following subsection), $[x_0]$ and $[\tau^*]$ were both divided into 10 equally wide subintervals, leading to a total of 100 interval simulations.

6.3 A Nonlinear System Model with Time-Varying Bounded Delay

As a third example, the model in Eq. (34) is reconsidered. Now, both the time delay and the state initialization function are assumed to be arbitrarily variable within

their respective interval bounds. Hence, only a single interval evaluation (instead of the 100 simulations from the previous subsection) was performed to obtain the exponential state enclosure. It can be noticed that the temporal variability of both of these quantities has only a minor influence on the solution enclosures because the interval bounds obtained by the exponential enclosure technique in Fig. 2(b) are only slightly wider than those in Fig. 2(a). However, as expected, the previous time-invariant case represents a subset of the solution to the time-varying scenario.

For the sake of comparison, a grid-based simulation is included in Fig. 2(b). It is based on the evaluation of the dynamic system model (34) with the help of the MATLAB routine `ddesd` with a maximum step size of $\underline{\tau}^* = 0.1$. In total, 2,000 equally distributed random sequences for the state initialization function and for the time delay were generated to mimic the influence of the uncertain quantities.

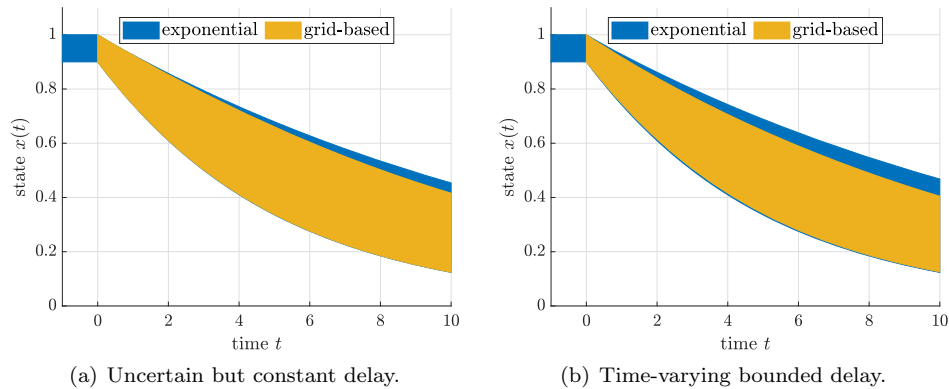


Figure 2: State enclosures for the linear system model (34) with uncertain delay.

For both Secs. 6.2 and 6.3, the use of the interval-based simulation approach has the advantage of a much smaller number of required system evaluations than in the grid-based counterpart, while further providing state enclosures that contain all reachable state values with certainty.

In Tab. 1, a comparison of the computing times⁷ for the grid-based floating point implementation using the routine `ddesd` and the novel exponential enclosure approach implemented with the help of the interval library INTLAB [31] (version 10.2) is given⁸. For identical discretization step sizes, and a grid-based simulation consisting of 2,000 individual runs, the exponential enclosure technique is faster by a factor of at least 175. Although the interval simulation has not been optimized for speed, it is faster by a factor of 8.77 even if the grid-based simulation is carried out with the largest investigated step size and the interval-based simulation with the smallest. In addition, it can be seen from Tab. 1 that a reduction of the

⁷All simulations were performed in MATLAB R2019b on a notebook computer under Windows 10, 64bit, 8 GB RAM, Intel Core i7-4500U CPU (@1.80GHz).

⁸Prototypical implementations are available for download on <https://github.com/ValEncIA-IVP/> and www.valencia-ivp.com.

discretization step size in the proposed approach leads to tighter interval bounds. A meaningful reduction of the computed interval diameters can be observed until $\Delta t = 0.01$ in the example considered in Sec. 6.3.

Table 1: Comparison of the grid-based simulation and the exponential enclosure technique for the example of Sec. 6.3.

step size	ddesd	proposed approach	speedup	diam $\{[x_e](10)\}$
$\Delta t = 0.1$	0.1272 s	0.9811 s	259.3	0.3462
$\Delta t = 0.01$	1.2297 s	12.474 s	197.1	0.3445
$\Delta t = 0.005$	2.5573 s	29.017 s	176.2	0.3444

6.4 Simulation of Wright's Equation with an Uncertain Parameter

A further nonlinear application scenario in this section illustrates a possible approach to circumvent cases in which the proposed simulation technique is not directly applicable due to a division by zero in the Theorems 2 and 3. We consider Wright's equation as published in [37]. Originally, it was formulated as

$$\dot{y}(t) = -p \cdot y(t-1) \cdot (1 + y(t)) \quad (35)$$

with the parameter $p > 0$. After the time-invariant change of coordinates

$$x(t) = 1 + y(t) \quad , \quad (36)$$

the equivalent formulation

$$\dot{x}(t) = -p \cdot (x(t-1) - 1) \cdot x(t) \quad (37)$$

is obtained, for which we assume a constant initialization function $x(t) = x_0$ for $t \leq 0$ with the consistent initial condition $x_0 = 2$ at the point $t = 0$ in the remainder of this subsection.

The change of coordinates (36) helps to avoid that solutions cross the value $y = 0$ if initialized with non-negative functions $y(t) > 0$ for $t \leq 0$ and positive parameters $p > 0$. The advantage of the exclusion of the solution $y = 0$ from the solution set is that a singularity in the iterations of Theorems 2 and 3 as well as Corollary 2 can be avoided.

For the Wright equation, this change of coordinates leads to a simplification of the iteration formula (23) according to

$$[\lambda_i]_m^{(\kappa+1)} := -p \cdot [x_e]_{m-1}([t] - \tau^*) \quad \text{with} \quad \tau^* = 1 \quad . \quad (38)$$

For the known parameter $p = 1$, Fig. 3 shows a comparison of the solution enclosures with the corresponding widths of the computed interval bounds for different discretization step sizes Δt . It can be seen clearly that the reduction in the

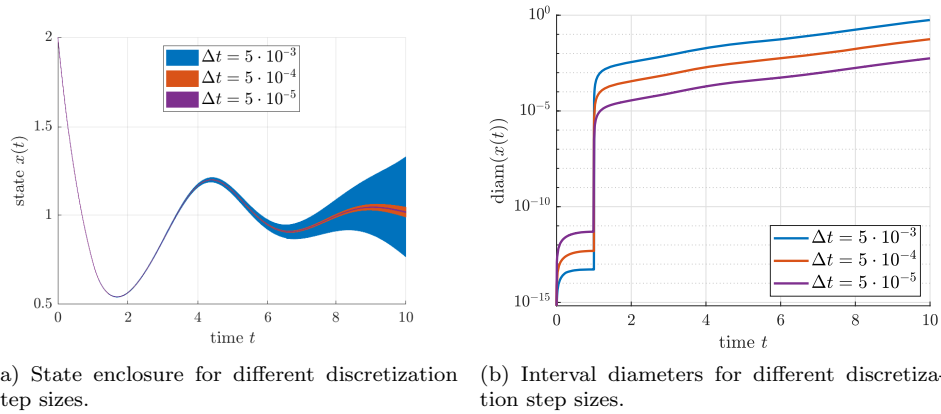


Figure 3: Simulation of the transformed Wright equation (37) for $p = 1$ with exactly known initialization function and initial condition.

interval widths is proportional to the reduction of the discretization step size, going along with a proportional increase of the computing time. It should be pointed out additionally that the reformulated iteration in (38) has the advantage for this specific benchmark scenario that its right-hand side is independent on the parameter to be computed and, hence, can be resolved explicitly by exploiting already pre-computed solution enclosures.

Using the step size $\Delta t = 5 \cdot 10^{-4}$, the simulation was repeated in Fig. 4 for the uncertain parameter interval $p \in [0.1 ; 2]$ in combination with equidistantly subdividing it into 100 subdomains. The resulting exponential enclosures tightly

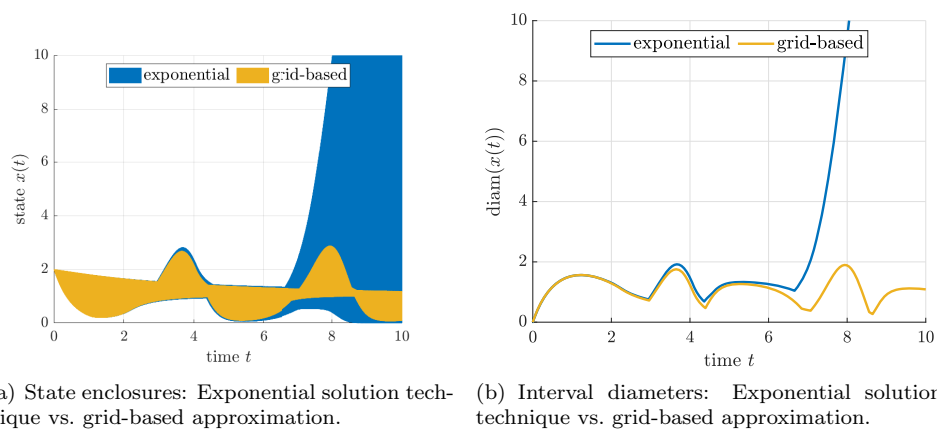


Figure 4: Simulation of the transformed Wright equation (37) for $p \in [0.1 ; 2]$ with exactly known initialization function and initial condition.

enclose a grid-based non-verified simulation performed with the routine `ddesd` up to $t \approx 6$. Afterwards, interval-related overestimation leads to a rapid inflation of the computed bounds. In future work, this can be countered by extending the complex-valued enclosure approach from [28, 29] to the case of scalar differential equations with delay. The oscillatory behavior of Wright's equation can then be better represented by not choosing purely real solution parameters $[\lambda_i]_m^{(\kappa+1)}$. For multi-dimensional systems, this approach is already implemented, see the simulation results in the following subsection.

6.5 Spring-Mass-Damper System

As a final application scenario, the oscillation attenuation of a spring-mass-damper system with the position variable x_1 , the velocity x_2 , and the actuating force x_3 is considered. The state equations

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \begin{bmatrix} 0 & 1 & 0 \\ p_1 \cdot a_{21} & p_2 \cdot a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix} \cdot \mathbf{x}(t) + \begin{bmatrix} 0 \\ 0 \\ b_3 \end{bmatrix} \cdot u(t) \\ &= \mathbf{A}(p_1, p_2) \cdot \mathbf{x}(t) + \mathbf{b} \cdot u(t) \end{aligned} \quad (39)$$

with a delay-free realization of the control input $u(t)$ were presented in [26, 27] as a benchmark scenario for the design of a robust output feedback controller in which the input $u(t)$ was chosen to be proportional to the velocity $x_2(t)$. From an engineering viewpoint, this model describes the simplest linear representation of an actively controlled wheel suspension system with a first-order lag behavior of the actuator. In the following, the control law is defined as

$$u(t) = 0.8 \cdot x_2(t - \tau^*) \quad , \quad (40)$$

where the constant gain factor 0.8 guarantees asymptotic stability of the nominal system with the parameters $a_{21} = -200$, $a_{22} = -15$, $a_{23} = -400$, $a_{33} = -200$, $b_3 = 10$, $p_1 = 1$, and $p_2 = 1$.

For the simulations in this subsection, we consider the cases of a delay-free system ($\tau^* = 0$), a relatively small delay ($\tau^* = 10^{-4}$) and a large delay ($\tau^* = 0.1$). In all scenarios, the integration step size is constant with $\Delta t = 10^{-5}$. Moreover, the independent parameters p_i , $i \in \{1, 2\}$, with their midpoints $p_{i,m} = 1$ are assumed to have the following identical bounds in the four cases shown in Figs. 5–7:

P1 $p_i = p_{i,m}$;

P2 $p_i \in p_{i,m} + [-0.005 ; 0.005]$;

P3 $p_i \in p_{i,m} + [-0.1 ; 0.1]$;

P4 $p_i \in p_{i,m} + [-0.5 ; 0.5]$.

In all cases, the system's initial conditions (and temporally constant initialization functions for $\tau^* > 0$) are defined as

$$\mathbf{x}_0 = [1 \quad 0 \quad 0.5]^T . \quad (41)$$

To reduce the influence of the wrapping effect, a time invariant change of coordinates is performed according to

$$\mathbf{z}(t) = \mathbf{V}^{-1} \cdot \mathbf{x}(t) , \quad (42)$$

where \mathbf{V} is the columnwise defined matrix of eigenvectors of

$$\mathbf{A}(p_{1,m}, p_{2,m}) + \mathbf{b} \cdot [0 \quad 0.8 \quad 0] \quad (43)$$

in the delay-free case and the eigenvector matrix of $\mathbf{A}(p_{1,m}, p_{2,m})$ in the case $\tau^* > 0$. For a non-zero delay, the change of coordinates leads to a complex-valued set of state equations as introduced in [28, 29].

In all simulations, it can be seen from the figures that the computed state enclosures are tight for the cases **P1** and **P2**. In contrast, the computed bounds start to inflate in **P4** for at least one of the state variables. To analyze this phenomenon with the help of suitable Lyapunov-Krasovskii functionals [8] is our future work. If the overall dynamics can be proven to be stable despite uncertain parameters and non-zero delay, Lyapunov-Krasovskii functionals can assist in excluding parts of the state enclosures that certainly do not belong to the reachable domains. This strategy can be interpreted as a generalization of the interval-based Lyapunov function technique presented in [12]. Moreover, such kind of analysis might help to distinguish the reasons for a blow-up of the computed enclosures. Possible causes are

- (a) the inflation due to excessively large discretization step sizes Δt ,
- (b) the inflation due to the wrapping effect that can be countered by splitting parameter intervals or performing a change of coordinates, or
- (c) the inflation of the bounds due to a destabilization of the system dynamics due to a large delay in the feedback control law (40).

Note that a point-valued simulation of the system considered in this section shows that the controlled model with the matrix $\sup(\mathbf{A}([p_1], [p_2]))$ in **P4** is unstable for $\tau^* \approx 0.103$ (and also further increased delays) which is only slightly larger than the delay considered in Fig. 7. This observation explains the rapid blow-up of the state enclosures in Figs. 7(j) and 7(k).

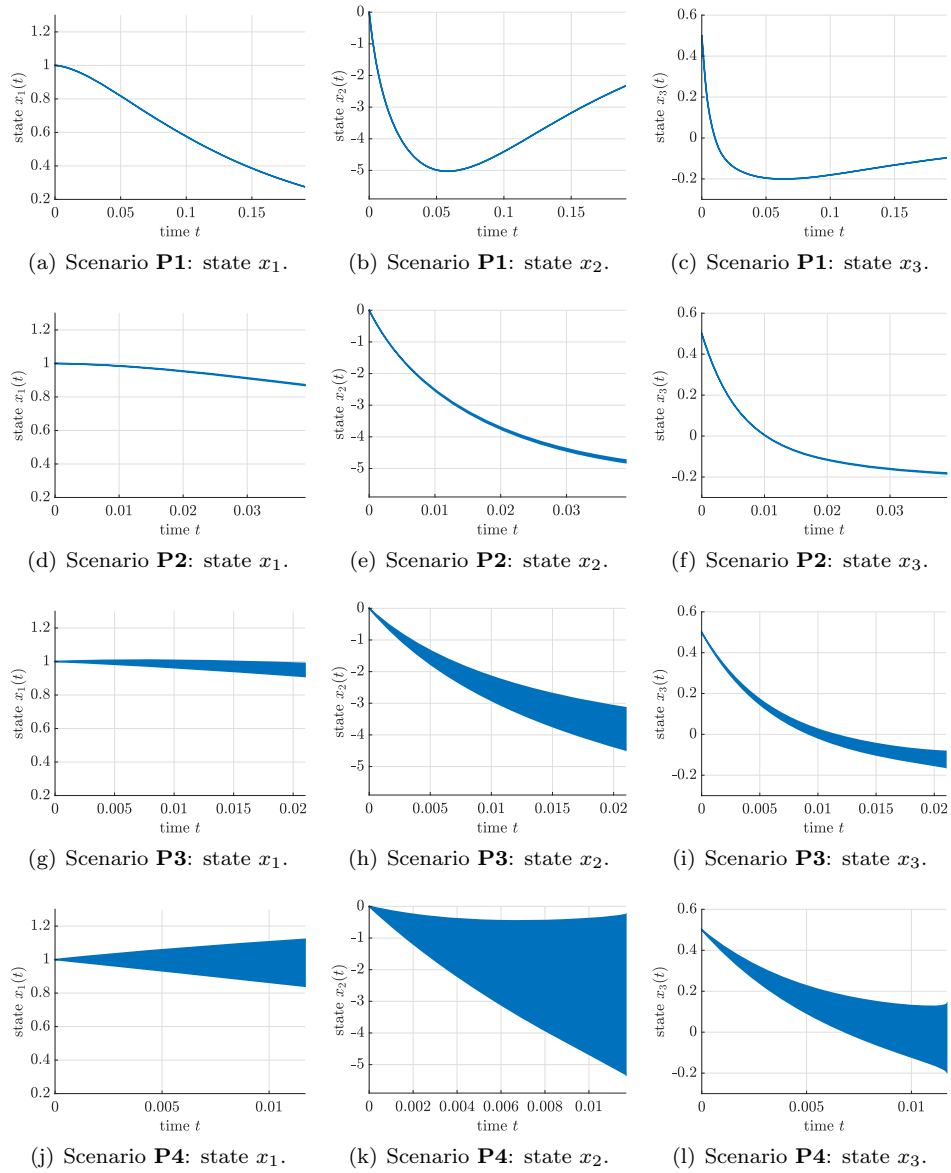


Figure 5: Simulation of the system model (39) for $\tau^* = 0$.

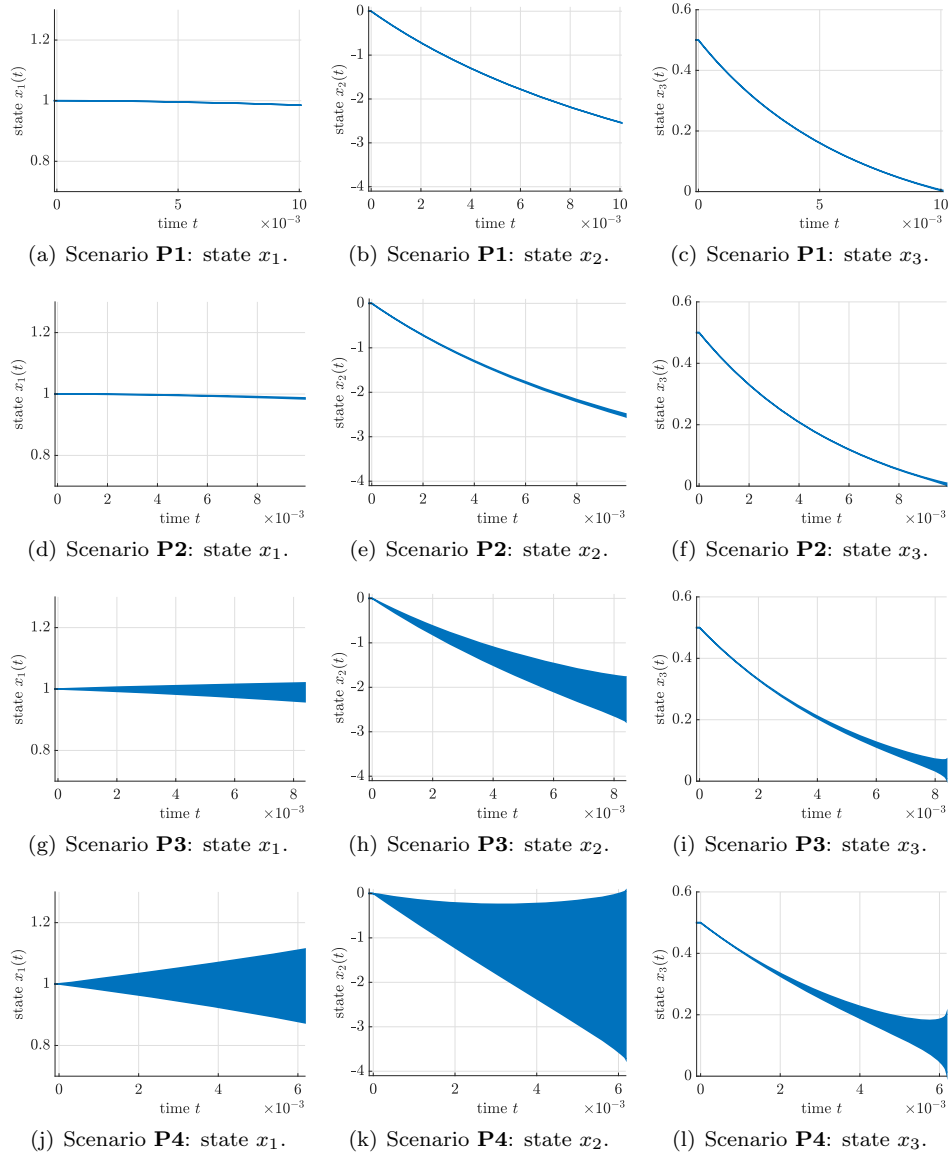
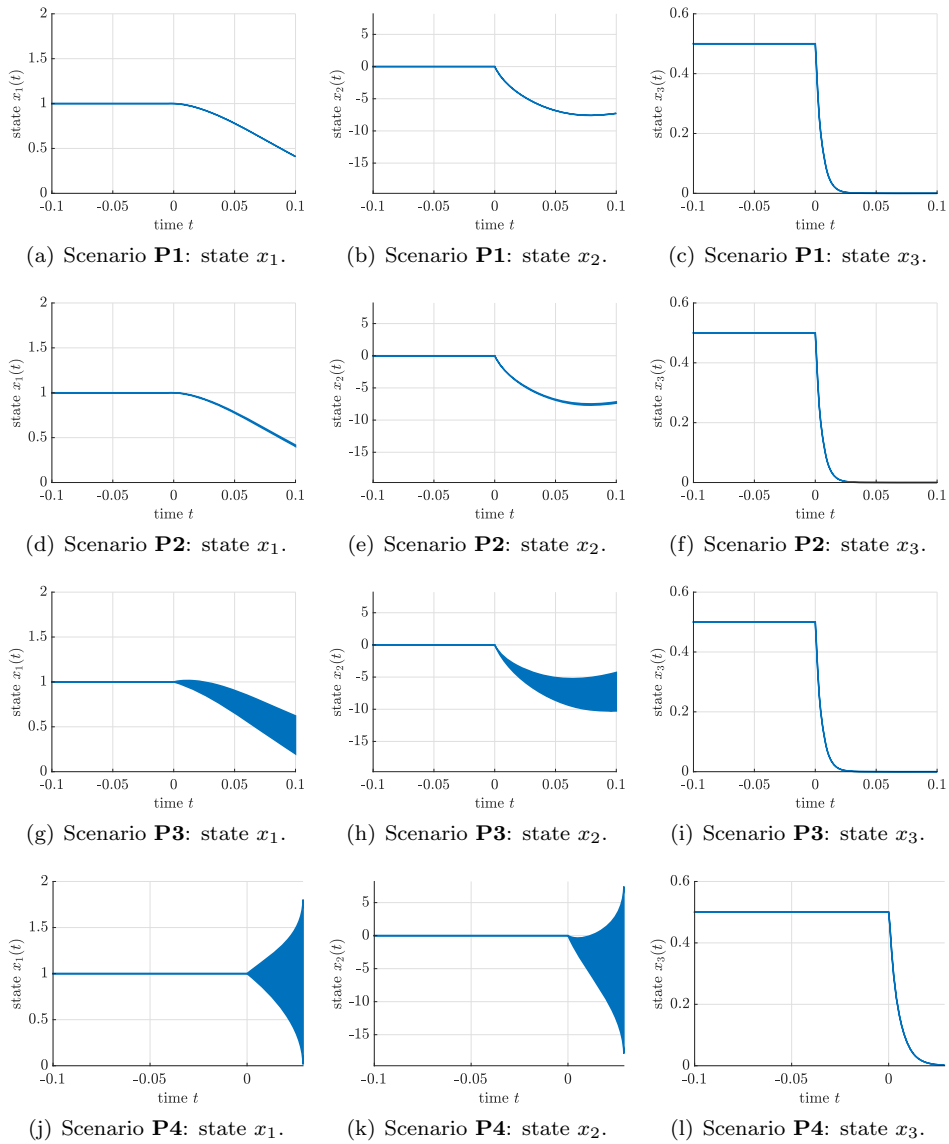


Figure 6: Simulation of the system model (39) for $\tau^* = 10^{-4}$.

Figure 7: Simulation of the system model (39) for $\tau^* = 0.1$.

In conclusion, we would like to point out the following fact. In many control engineering applications, it is possible to intersect the computed bounds with information from measurements at discrete time instants. In practice, this additional information allows us to work even with wide parameter bounds to forecast the domains of reachable states in a computationally cheap manner by providing simple enclosures between two distinct measurement points. This well-known predictor-corrector concept can be implemented using the approach suggested in this paper even within real-time capable state estimation and (model-predictive) control frameworks. The predictor-corrector idea with continuous dynamics and discrete-time measurements is based on the so-called hybrid Kalman filter for systems with stochastic uncertainty [34].

7 Conclusions and Future Work

In this paper, a novel interval-based solution method for certain classes of delay equations was presented. It extends an exponential enclosure technique that was originally developed for delay-free systems of ordinary differential equations.

As future work, we plan to extend the exponential enclosure technique to fractional-order differential equations by replacing the exponential terms with so-called Mittag-Leffler functions [23, 24]. Fractional-order models have a large practical relevance in the context of control and state estimation of electrochemical energy converters and storage elements such as fuel cells and batteries. Moreover, extensions of the proposed technique for solving delay differential equations to systems including the value zero in the set of reachable states will be investigated. Our goal will be to find non-trivial alternatives based on the complex-valued iteration technique published in [29] with the focus on oscillatory dynamics.

References

- [1] Arcara, P. and Melchiorri, C. Control schemes for teleoperation with time delay: A comparative study. *Robotics and Autonomous Systems*, 38(1):49–64, 2002. DOI: 10.1016/S0921-8890(01)00164-6.
- [2] Auer, E., Rauh, A., and Kersten, J. Experiments-based parameter identification on the GPU for cooperative systems. *Journal of Computational and Applied Mathematics*, 371:112657, 2019. DOI: 10.1016/j.cam.2019.112657.
- [3] Cooke, K.L. Stability of delay differential equations with applications in biology and medicine. In Capasso, V., Grosso, E., and Paveri-Fontana, S.L., editors, *Mathematics in Biology and Medicine*, pages 439–446, Berlin, Heidelberg, 1985. Springer. DOI: 10.1007/978-3-642-93287-8_59.
- [4] Csendes, T. and Ratz, D. Subdivision direction selection in interval methods for global optimization. *SIAM Journal on Numerical Analysis*, 34(3):922–938, 1997. DOI: 10.1137/S0036142995281528.

- [5] Driver, R.D. *Ordinary and Delay Differential Equations*. Springer-Verlag, New York, 1977. DOI: 10.1007/978-1-4684-9467-9.
- [6] Erneux, T. *Applied Delay Differential Equations*. Springer-Verlag, New York, 2009. DOI: 10.1007/978-0-387-74372-1.
- [7] Fridman, E. *Introduction to Time-Delay Systems: Analysis and Control*. Birkhäuser, Basel, 2014. DOI: 10.1007/978-3-319-09393-2.
- [8] Fridman, E. Tutorial on Lyapunov-based methods for time-delay systems. *European Journal of Control*, 20(6):271–283, 2014. DOI: 10.1016/j.ejcon.2014.10.001.
- [9] Gopalsamy, K. *Stability and Oscillations in Delay Differential Equations of Population Dynamics*. Springer-Verlag, Dordrecht, 1992.
- [10] Groothedde, C. and Mireles James, J. Parameterization method for unstable manifolds of delay differential equations. *Journal of Computational Dynamics*, 4(1&2):21–70, 2017. DOI: 10.3934/jcd.2017002.
- [11] Jaulin, L., Kieffer, M., Didrit, O., and Walter, É. *Applied Interval Analysis*. Springer-Verlag, London, 2001. DOI: 10.1007/978-1-4471-0249-6.
- [12] Kersten, J., Rauh, A., and Aschemann, H. Interval methods for robust gain scheduling controllers: An LMI-based approach. *Granular Computing*, 5:203–216, 2020. DOI: 10.1007/s41066-018-00147-1.
- [13] Kyrychko, Y. and Hogan, S. On the use of delay equations in engineering applications. *Journal of Vibration and Control*, 16(7–8):943–960, 2010. DOI: 10.1177/1077546309341100.
- [14] Lin, Y. and Stadtherr, M.A. Validated solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, 57(10):1145–1162, 2007. DOI: 10.1016/j.apnum.2006.10.006.
- [15] Lohner, R. Enclosing the solutions of ordinary initial and boundary value problems. In Kaucher, E.W., Kulisch, U.W., and Ullrich, C., editors, *Computer Arithmetic: Scientific Computation and Programming Languages*, pages 255–286, Stuttgart, 1987. Wiley-Teubner Series in Computer Science.
- [16] Lohner, R. On the ubiquity of the wrapping effect in the computation of the error bounds. In Kulisch, U., Lohner, R., and Facius, A., editors, *Perspectives on Enclosure Methods*, pages 201–217, Wien, New York, 2001. Springer-Verlag. DOI: 10.1007/978-3-7091-6282-8_12.
- [17] Martins, A., Laranjeira, P., Dias, M., and Lopes, J.C. Mass transport modelling in porous media using delay differential equations. *Defect and Diffusion Forum*, 258:586–591, 2006. DOI: 10.4028/www.scientific.net/DDF.258-260.586.

- [18] Michiels, W. and Niculescu, S.I. *Stability and Stabilization of Time-delay Systems: An Eigenvalue-based Approach*. Advances in Design and Control. SIAM, Philadelphia, PA, 2007. DOI: 10.1137/1.9780898718645.
- [19] Moore, R.E., Kearfott, R.B., and Cloud, M.J. *Introduction to Interval Analysis*. SIAM, Philadelphia, 2009. DOI: 10.1137/1.9780898717716.
- [20] Nedialkov, N.S. Interval tools for ODEs and DAEs. In *CD-Proceedings of the 12th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics SCAN 2006*, Duisburg, Germany, 2007. IEEE Computer Society. DOI: 10.1109/SCAN.2006.28.
- [21] Olgac, N., Jenkin, R., and Zalluhoglu, U. A practical approach to the complete stability of TDS with multiple identical imaginary roots. In *Proceedings of the 18th European Control Conference ECC 2020*, St. Petersburg, Russia, 2020. DOI: 10.23919/ECC51009.2020.9143924.
- [22] Rauh, A. *Sensitivity Methods for Analysis and Design of Dynamic Systems with Applications in Control Engineering*. Shaker-Verlag, 2017. DOI: 10.2370/9783844054989.
- [23] Rauh, A. and Jaulin, L. Novel techniques for a verified simulation of fractional-order differential equations. *Fractal and Fractional*, 5(1):17, 2021. DOI: 10.3390/fractalfract5010017.
- [24] Rauh, A. and Kersten, J. Toward the development of iteration procedures for the interval-based simulation of fractional-order systems. *Acta Cybernetica*, 24(3):343–372, 2020. DOI: 10.14232/actacyb.285660.
- [25] Rauh, A., Kersten, J., and Aschemann, H. Techniques for verified reachability analysis of quasi-linear continuous-time systems. In *Proceedings of the 24th International Conference on Methods and Models in Automation and Robotics, Miedzydroje, Poland*, 2019. DOI: 10.1109/MMAR.2019.8864648.
- [26] Rauh, A. and Romig, S. Linear matrix inequalities for an iterative solution of robust output feedback control of systems with bounded and stochastic uncertainty. *Sensors*, 21(9):3285, 2021. DOI: 10.3390/s21093285.
- [27] Rauh, A., Romig, S., and Aschemann, H. When is naive low-pass filtering of noisy measurements counter-productive for the dynamics of controlled systems? In *Proceedings of the 23rd IEEE International Conference on Methods and Models in Automation and Robotics MMAR 2018*, Miedzydroje, Poland, 2018. DOI: 10.1109/MMAR.2018.8486099.
- [28] Rauh, A., Westphal, R., and Aschemann, H. Verified Simulation of Control Systems with Interval Parameters Using an Exponential State Enclosure Technique. In *CD-Proceedings of the IEEE International Conference on Methods and Models in Automation and Robotics MMAR*, Miedzydroje, Poland, 2013. DOI: 10.1109/MMAR.2013.6669913.

- [29] Rauh, A., Westphal, R., Aschemann, H., and Auer, E. Exponential enclosure techniques for initial value problems with multiple conjugate complex eigenvalues. In Nehmeier, M., Wolff von Gudenberg, J., and Tucker, W., editors, *Scientific Computing, Computer Arithmetic, and Validated Numerics*, pages 247–256, Cham, 2016. Springer International Publishing. DOI: 10.1007/978-3-319-31769-4_20.
- [30] Roussel, Marc R. *Nonlinear Dynamics: A Hands-On Introductory Survey*. Morgan & Claypool Publishers, 2019. DOI: 10.1088/2053-2571/ab0281.
- [31] Rump, S.M. INTLAB — INTerval LABoratory. In Csendes, T., editor, *Developments in Reliable Computing*, pages 77–104. Kluwer Academic Publishers, 1999. DOI: 10.1007/978-94-017-1247-7_7.
- [32] Seuret, A., Hetel, L., Daafouz, J., and Johansson, K.H. *Delays and Networked Control Systems*. Advances in Delays and Dynamics. Springer International Publishing, Switzerland, 2016. DOI: 10.1007/978-3-319-32372-5.
- [33] Shampine, L.F. and Thompson, S. Numerical solution of delay differential equations. In Gilsinn, D.E., Kalmár-Nagy, T., and Balachandran, B., editors, *Delay Differential Equations: Recent Advances and New Directions*, pages 245–271. Springer-Verlag, Boston, MA, 2009. DOI: 10.1007/978-0-387-85595-0_9.
- [34] Stengel, R. *Optimal Control and Estimation*. Dover Publications, Inc., 1994.
- [35] Szczelina, R. and Zgliczynski, P. Algorithm for rigorous integration of delay differential equations and the computer-assisted proof of periodic orbits in the Mackey–Glass equation. *Foundations of Computational Mathematics*, 18:1299–1332, 2018. DOI: 10.1007/s10208-017-9369-5.
- [36] van den Berg, J.B., Groothedde, C., and Lessard, J.-P. A general method for computer-assisted proofs of periodic solutions in delay differential problems. *Journal of Dynamics and Differential Equations*, pages 1–44, 2020. DOI: 10.1007/s10884-020-09908-6.
- [37] Wright, E.M. The Non-Linear Difference-Differential Equation. *The Quarterly Journal of Mathematics*, os-17(1):245–252, 1946. DOI: 10.1093/qmath/os-17.1.245.

Received 29th October 2020

The Generalized Epsilon Function: An Alternative to the Exponential Function

Tamás Jónás^a

Abstract

It is well known that the exponential function plays an extremely important role in many areas of science. In this study, a generator function-based mapping, called the generalized epsilon function is presented. Next, we demonstrate that the exponential function is an asymptotic generalized epsilon function. Exploiting this result and the fact that this new function is generator function-dependent, it can be utilized as a very flexible alternative to the exponential function in a wide range of applications. We should add that if the generator is a rational function, then the generalized epsilon function is rational as well. In this case, the generalized epsilon function is computationally simple and it may be treated as an easy-to-compute alternative to the exponential function. In this paper, we briefly present two applications of this novel function: an approximation to the exponential probability distribution, and an alternative to the sigmoid function on a bounded domain.

Keywords: exponential function, approximation, epsilon function, exponential distribution, sigmoid function

1 Preliminaries

In [9], Dombi *et al.* introduced the epsilon function and by using this mapping the authors constructed the epsilon probability distribution that may be viewed as an alternative to the exponential probability distribution. The epsilon function is defined as follows.

Definition 1. *The epsilon function $\varepsilon_d^{(\lambda)}(x)$ is given by*

$$\varepsilon_d^{(\lambda)}(x) = \left(\frac{d+x}{d-x} \right)^{\lambda \frac{d}{2}}, \quad (1)$$

where $\lambda \in \mathbb{R} \setminus \{0\}$, $d > 0$, $x \in (-d, +d)$.

^aFaculty of Economics, Eötvös Loránd University, Rákóczi út 7, H-1088 Budapest, Hungary, E-mail: jonas@gtk.elte.hu, ORCID: 0000-0001-8241-2321

In [9], we proved the following theorem, which states an important asymptotic property of the epsilon function.

Theorem 1. *For any $x \in (-d, +d)$, if $d \rightarrow \infty$, then*

$$\varepsilon_d^{(\lambda)}(x) \rightarrow e^{\lambda x}. \quad (2)$$

It should be mentioned that this results was also utilized for constructing an effective approximation to the normal probability distribution (see [8]).

The epsilon function given in Definition 1 may be treated as an alternative of the exponential function on the domain $(-d, d)$. The exponential function has an extremely wide range of applications in many areas of science including mathematics, physics, chemistry, computer science, economics and biology (see, e.g., [19, 1, 4, 15, 5]). Our motivation was to generalize the epsilon function such that it can be used to approximate the exponential function with an even higher level of flexibility.

In this paper, we will present the generalized epsilon function, which is a generator function-based mapping from the domain $[-d, d]$ to the non-negative extended real line ($d > 0$). We will prove that if $d \rightarrow \infty$, then the generalized epsilon function coincides with the exponential function. This result allows us to treat the generalized epsilon function as an alternative to exponential function on a bounded domain. Since this new function is generator function-dependent, it is very flexible and it can be utilized in a wide range of applications. Here, we will briefly present two applications of the generalized epsilon function: an approximation to the exponential probability distribution, and an alternative to the sigmoid function on a bounded domain.

We will use the common notation \mathbb{R} for the real line and $\overline{\mathbb{R}}$ for the extended real line, i.e., $\overline{\mathbb{R}} = [-\infty, \infty]$. Also, $\overline{\mathbb{R}}_+$ will denote the non-negative extended real line, i.e., $\overline{\mathbb{R}}_+ = [0, \infty]$. We will consider the arithmetic operations on the extended real line according to Klement *et al.* [13] and Grabisch *et al.* [10].

2 The generalized epsilon function

First, we will construct a generator function-based mapping, called the general epsilon function, which we can use to approximate the exponential function. Then, we will prove that the exponential function is an asymptotic generalized epsilon function.

2.1 Construction

Let $d \in \mathbb{R}$, $d > 0$ and let $\lambda \in \mathbb{R} \setminus \{0\}$. Our aim is to construct a function $f_d^{(\alpha)}: [-d, d] \rightarrow \overline{\mathbb{R}}_+$ in the form

$$f_d^{(\alpha)}(x) = c h_d^\alpha(x), \quad (3)$$

where $h_d: [-d, d] \rightarrow \overline{\mathbb{R}}_+$ is a continuous and strictly monotonic mapping, $c > 0$ and $\alpha \in \mathbb{R} \setminus \{0\}$, such that $f_d^{(\alpha)}$ approximates the exponential function $e^{\lambda x}$ on the domain $[-d, d]$. Noting the basic properties of the exponential function $e^{\lambda x}$, we set the following requirements for $f_d^{(\alpha)}$:

- (a) For any $x \in [-d, d]$, $f_d^{(\alpha)}(x) \in \overline{\mathbb{R}}_+$.
- (b) If $\lambda > 0$ ($\lambda < 0$, respectively), then $f_d^{(\alpha)}$ is strictly increasing (decreasing, respectively).
- (c) $f_d^{(\alpha)}$ is differentiable on $(-d, d)$; and $f_d^{(\alpha)}(x)$ and $e^{\lambda x}$ are identical to first order at $x = 0$, i.e.,

(c1)

$$f_d^{(\alpha)}(0) = 1 \quad \text{and}$$

(c2)

$$\left. \frac{df_d^{(\alpha)}(x)}{dx} \right|_{x=0} = \lambda.$$

As $c > 0$ and $\alpha \neq 0$, there exists a $\hat{c} > 0$ such that $c = \hat{c}^\alpha$. Using this substitution, Eq. (3) can be written as

$$f_d^{(\alpha)}(x) = (\hat{c} h_d(x))^\alpha.$$

Since we wish $f_d^{(\alpha)}$ to be a generator function-based mapping from $[-d, d]$ to $\overline{\mathbb{R}}_+$ (see requirement (a)), let h_d have the form

$$h_d(x) = g\left(\frac{x+d}{2d}\right), \quad x \in [-d, d],$$

where $g: [0, 1] \rightarrow \overline{\mathbb{R}}_+$ is a continuous and strictly monotonic function. This means that

$$f_d^{(\alpha)}(x) = \left(\hat{c} g\left(\frac{x+d}{2d}\right)\right)^\alpha \tag{4}$$

for any $x \in [-d, d]$. We will call the function g the generator of $f_d^{(\alpha)}$.

Taking into account requirement (b), we have that if $\lambda > 0$, then for a strictly increasing (decreasing, respectively) generator g , α has to be positive (negative, respectively). Similarly, noting requirement (b), we have that if $\lambda < 0$, then for a strictly increasing (decreasing, respectively) generator g , α has to be negative (positive, respectively).

Using Eq. (4), the requirement (c1) leads us to

$$\left(\hat{c} g\left(\frac{1}{2}\right)\right)^\alpha = 1,$$

from which

$$\hat{c} = \left(g \left(\frac{1}{2} \right) \right)^{-1}$$

and so Eq. (4) can be written as

$$f_d^{(\alpha)}(x) = \left(\frac{g \left(\frac{x+d}{2d} \right)}{g \left(\frac{1}{2} \right)} \right)^\alpha \quad (5)$$

for any $x \in [-d, d]$.

Next, considering requirement (c2), we get that g has to be a differentiable function on $(0, 1)$ and

$$\left(f_d^{(\alpha)}(x) \right)' \Big|_{x=0} = \left(\left(\frac{g \left(\frac{x+d}{2d} \right)}{g \left(\frac{1}{2} \right)} \right)^\alpha \right)' \Big|_{x=0} = \lambda. \quad (6)$$

The first derivative of $f_d^{(\alpha)}$ is

$$\left(f_d^{(\alpha)}(x) \right)' = \alpha \left(\frac{g \left(\frac{x+d}{2d} \right)}{g \left(\frac{1}{2} \right)} \right)^{\alpha-1} \frac{g' \left(\frac{x+d}{2d} \right)}{g \left(\frac{1}{2} \right)} \frac{1}{2d}$$

and so from Eq. (6), we get

$$\alpha = 2\lambda d \frac{g \left(\frac{1}{2} \right)}{g' \left(\frac{1}{2} \right)}.$$

Hence, using Eq. (5), $f_d^{(\alpha)}(x)$ can be written as

$$f_d^{(\alpha)}(x) = \left(\frac{g \left(\frac{x+d}{2d} \right)}{g \left(\frac{1}{2} \right)} \right)^{2\lambda d \frac{g \left(\frac{1}{2} \right)}{g' \left(\frac{1}{2} \right)}}$$

for any $x \in [-d, d]$. Notice that the generator function g needs to meet the criterion $g' \left(\frac{1}{2} \right) \neq 0$ as well.

Remark 1. If $\lambda > 0$ and g is strictly increasing, then $g' \left(\frac{1}{2} \right) > 0$ and so $\alpha > 0$, which means that $f_d^{(\alpha)}$ is strictly increasing on $[-d, d]$. Similarly, if $\lambda > 0$ and g is strictly decreasing, then $g' \left(\frac{1}{2} \right) < 0$, which implies $\alpha < 0$ and so $f_d^{(\alpha)}$ is strictly increasing on $[-d, d]$. Therefore, if $\lambda > 0$, then $f_d^{(\alpha)}$ is strictly increasing on $[-d, d]$ regardless if g is a strictly increasing or a strictly decreasing function. Based on similar considerations, we get that if $\lambda < 0$, then $f_d^{(\alpha)}$ is strictly decreasing on $[-d, d]$ independently of the monotonicity of function g . Therefore, $f_d^{(\alpha)}$ satisfies the requirement (b).

Now, using the construction presented so far, we will introduce the so-called generalized epsilon function, which can be used to approximate the exponential function. In this definition, we will utilize the following class of functions.

Definition 2. Let \mathcal{G} be the set of all functions $g: [0, 1] \rightarrow \overline{\mathbb{R}}_+$ that are strictly monotonic and differentiable on $(0, 1)$ with $g'(\frac{1}{2}) \neq 0$, where g' denotes the first derivative of g , and g' is continuous on $(0, 1)$.

For a strictly increasing $g \in \mathcal{G}$, $g(1) = \infty$ will mean the limit $\lim_{x \rightarrow 1} g(x) = \infty$. Similarly, for a strictly decreasing $g \in \mathcal{G}$, $g(0) = \infty$ will stand for the limit $\lim_{x \rightarrow 0} g(x) = \infty$.

Definition 3 (Generalized epsilon function). Let $g \in \mathcal{G}$, let $\lambda \in \mathbb{R} \setminus \{0\}$ and $d > 0$. We say that the function $\varepsilon_{d,g}^{(\lambda)}: [-d, d] \rightarrow \overline{\mathbb{R}}_+$, which is given by

$$\varepsilon_{d,g}^{(\lambda)}(x) = \left(\frac{g\left(\frac{x+d}{2d}\right)}{g\left(\frac{1}{2}\right)} \right)^{2\lambda d \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)}}, \tag{7}$$

is a generalized epsilon function (GEF) with the parameters λ and d induced by the generator function g .

Later, we will show that the epsilon function, which was first introduced by Dombi *et al.* in [9], is just a special case of function $\varepsilon_{d,g}^{(\lambda)}$. That is, $\varepsilon_{d,g}^{(\lambda)}$ may be viewed as a generalization of the epsilon function.

2.2 Identity of two generalized epsilon functions

As a GEF is generator function-dependent, the question when two GEFs are identical naturally arises. The following proposition gives a sufficient condition for the equality of two generalized epsilon functions that are induced by two generator functions.

Proposition 1. The GEF is uniquely determined up to any transformation

$$t(x) = \alpha x^\beta, \quad x \in \overline{\mathbb{R}}_+ \tag{8}$$

on its generator function, if $\alpha > 0$ and $\beta \in \mathbb{R} \setminus \{0\}$.

Proof. Let $\lambda \in \mathbb{R} \setminus \{0\}$, $d > 0$ and let the GEF $\varepsilon_{d,g}^{(\lambda)}$ be induced by the generator function $g \in \mathcal{G}$. Furthermore, let $\alpha > 0$ and $\beta \in \mathbb{R} \setminus \{0\}$. Now, let the function $t: \overline{\mathbb{R}}_+ \rightarrow \overline{\mathbb{R}}_+$ be given by Eq. (8), and let $h(x) = t(g(x))$ for any $x \in [0, 1]$. Then, $h \in \mathcal{G}$ and for any $x \in [-d, d]$, the GEF induced by h can be written as

$$\begin{aligned} \varepsilon_{d,h}^{(\lambda)}(x) &= \left(\frac{h\left(\frac{x+d}{2d}\right)}{h\left(\frac{1}{2}\right)} \right)^{2\lambda d \frac{h\left(\frac{1}{2}\right)}{h'\left(\frac{1}{2}\right)}} = \left(\frac{\alpha g^\beta\left(\frac{x+d}{2d}\right)}{\alpha g^\beta\left(\frac{1}{2}\right)} \right)^{2\lambda d \frac{\alpha g^\beta\left(\frac{1}{2}\right)}{\alpha \beta g^{\beta-1}\left(\frac{1}{2}\right)g'\left(\frac{1}{2}\right)}} = \\ &= \left(\frac{g\left(\frac{x+d}{2d}\right)}{g\left(\frac{1}{2}\right)} \right)^{2\lambda d \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)}} = \varepsilon_{d,g}^{(\lambda)}(x). \end{aligned}$$

□

3 The exponential function as an asymptotic generalized epsilon function

Let $\varepsilon_{d,g}^{(\lambda)}$ be a GEF induced by the generator function $g \in \mathcal{G}$, where $\lambda \in \mathbb{R} \setminus \{0\}$ and $d > 0$. Due to the construction of $\varepsilon_{d,g}^{(\lambda)}$, it has the following properties:

- For any $x \in [-d, d]$, $\varepsilon_{d,g}^{(\lambda)}(x) \in \overline{\mathbb{R}}_+$.
- If $\lambda > 0$ ($\lambda < 0$, respectively), then $\varepsilon_{d,g}^{(\lambda)}$ is strictly increasing (decreasing, respectively).
- $\varepsilon_{d,g}^{(\lambda)}(x)$ and $e^{\lambda x}$ are identical to first order at $x = 0$.

Here, we will demonstrate an important asymptotic property of the generalized epsilon function.

Theorem 2. *Let $g \in \mathcal{G}$, $\lambda \in \mathbb{R} \setminus \{0\}$ and $d > 0$. Let $\varepsilon_{d,g}^{(\lambda)}: [-d, d] \rightarrow \overline{\mathbb{R}}_+$ be a GEF induced by g according to Eq. (7). Then, for any $x \in (-d, +d)$,*

$$\lim_{d \rightarrow \infty} \varepsilon_{d,g}^{(\lambda)}(x) = e^{\lambda x}. \quad (9)$$

Proof. Using the definition of $\varepsilon_{d,g}^{(\lambda)}$, for any $x \in (-d, +d)$, Eq. (9) is equivalent to

$$\lim_{d \rightarrow \infty} \left(2\lambda d \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)} \ln \left(\frac{g\left(\frac{x+d}{2d}\right)}{g\left(\frac{1}{2}\right)} \right) \right) = \lambda x. \quad (10)$$

The left hand side of Eq. (10) can be written as

$$2\lambda \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)} \lim_{d \rightarrow \infty} \left(\frac{\ln \left(\frac{g\left(\frac{x+d}{2d}\right)}{g\left(\frac{1}{2}\right)} \right)}{\frac{1}{d}} \right). \quad (11)$$

Notice that we can use the L'Hospital rule to compute the limit in Eq. (11). Taking into account that g is differentiable on $(0, 1)$ and g' is continuous on $(0, 1)$, after direct calculation, we get

$$\begin{aligned} 2\lambda \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)} \lim_{d \rightarrow \infty} \left(\frac{\ln \left(\frac{g\left(\frac{x+d}{2d}\right)}{g\left(\frac{1}{2}\right)} \right)}{\frac{1}{d}} \right) &= 2\lambda \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)} \lim_{d \rightarrow \infty} \left(\frac{\left(\ln \left(\frac{g\left(\frac{x+d}{2d}\right)}{g\left(\frac{1}{2}\right)} \right) \right)'}{\left(\frac{1}{d}\right)'} \right) = \\ &= 2\lambda \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)} \lim_{d \rightarrow \infty} \left(\frac{g'\left(\frac{x+d}{2d}\right) x}{g\left(\frac{x+d}{2d}\right) 2} \right) = \lambda x. \end{aligned}$$

□

Example 1. Let $g(x) = \cos(x)$, where $x \in [0, 1]$. Clearly, $g \in \mathcal{G}$, i.e., g satisfies the requirements for a generator function of a generalized epsilon function. After direct calculation, we get that the GEF induced by the function g is

$$\varepsilon_{d,g}^{(\lambda)}(x) = \varepsilon_{d,g}^{(\lambda)}(x) = \left(\frac{\cos\left(\frac{x+d}{2d}\right)}{\cos\left(\frac{1}{2}\right)} \right)^{-\frac{2\lambda d}{\tan\left(\frac{1}{2}\right)}} \approx \left(1.14 \cdot \cos\left(\frac{x+d}{2d}\right) \right)^{-3.66 \cdot \lambda d},$$

where $d > 0$ and $x \in [-d, d]$. Table 1 shows the maximum absolute relative errors, i.e.,

$$\max_{x \in (-\Delta, \Delta)} \left| \frac{\varepsilon_{d,g}^{(\lambda)}(x) - e^{\lambda x}}{e^{\lambda x}} \right|,$$

of this approximation for various values of d and Δ , where $\lambda = 1$.

Table 1: The maximum absolute relative errors of the approximations of $e^{\lambda x}$ using $\varepsilon_{d,g}^{(\lambda)}(x)$, for $\lambda = 1$ and $g(x) = \cos(x)$.

d	$x \in (-2, 2)$	$x \in (-5, 5)$	$x \in (-10, 10)$	$x \in (-20, 20)$
100	2.42×10^{-2}	1.62×10^{-1}	8.32×10^{-1}	$1.08 \times 10^{+1}$
1000	2.38×10^{-3}	1.50×10^{-2}	6.13×10^{-2}	2.69×10^{-1}

Based on Table 1, the GEF approximates the exponential function around zero quite well, which is in line with the construction of $\varepsilon_{d,g}^{(\lambda)}$. On the other hand, if $x \gg 0$ or $x \ll 0$, ($x \in [-d, d]$), then the goodness of this approximation considerably decreases. Figure 1 shows the plots of the GEF, exponential function and their absolute relative difference for $\lambda = 1$, $d = 100$, $g(x) = \cos(x)$.

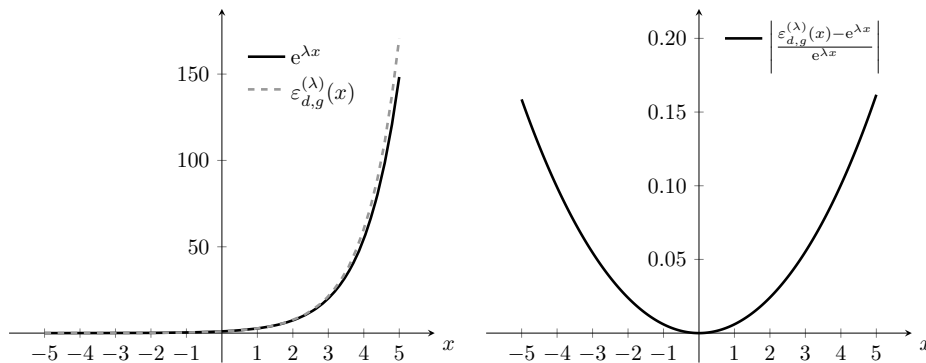


Figure 1: Plots of the GEF, exponential function and their absolute relative difference for $\lambda = 1$, $d = 100$, $g(x) = \cos(x)$.

We can achieve more effective approximations to the exponential function $e^{\lambda x}$ on the interval $[-d, d]$, if for a strictly increasing g we require $g(0) = 0$ and for a strictly decreasing g we require $g(1) = 0$. Table 2 summarizes the main properties of a GEF induced by g in these cases. Note that if g is a strictly increasing function with $g(0) = 0$ and $\lambda < 0$, then we interpret $\varepsilon_{d,g}^{(\lambda)}(-d)$ as $\varepsilon_{d,g}^{(\lambda)}(-d) = \lim_{x \rightarrow -d^+} \varepsilon_{d,g}^{(\lambda)}(x) = \infty$. Similarly, if g is a strictly decreasing function with $g(1) = 0$ and $\lambda > 0$, then we interpret $\varepsilon_{d,g}^{(\lambda)}(d)$ as $\varepsilon_{d,g}^{(\lambda)}(d) = \lim_{x \rightarrow d^-} \varepsilon_{d,g}^{(\lambda)}(x) = \infty$. In Table 2, $< \infty$ stands for a finite value, while \nearrow (\searrow , respectively) denotes that a function is strictly increasing (decreasing, respectively).

Table 2: Main properties of the GEF $\varepsilon_{d,g}^{(\lambda)}$ depending on the values of $g(0)$ and $g(1)$.

g	λ	$g(0)$	$g(1)$	$\varepsilon_{d,g}^{(\lambda)}(-d)$	$\varepsilon_{d,g}^{(\lambda)}(d)$	$\varepsilon_{d,g}^{(\lambda)}$
\nearrow	> 0	0	$< \infty$ (∞)	0	$< \infty$ (∞)	\nearrow
\nearrow	< 0	0	$< \infty$ (∞)	∞	> 0 (0)	\searrow
\searrow	> 0	$< \infty$ (∞)	0	> 0 (0)	∞	\nearrow
\searrow	< 0	$< \infty$ (∞)	0	$< \infty$ (∞)	0	\searrow

Example 2. Let $g(x) = x$, $x \in [0, 1]$. Then, $g(\frac{1}{2}) = \frac{1}{2}$, $g'(\frac{1}{2}) = 1$ and via direct calculation, we get that the GEF induced by g is

$$\varepsilon_{d,g}^{(\lambda)}(x) = \left(1 + \frac{x}{d}\right)^{\lambda d},$$

where $d > 0$ and $x \in [-d, d]$. It is well known that $\lim_{d \rightarrow \infty} \left(1 + \frac{x}{d}\right)^{\lambda d} = e^{\lambda x}$, which is in line with the result of Theorem 2.

It should be added that $\varepsilon_{d,g}^{(\lambda)}(x)$ is closely related to the cumulative distribution function of the p -exponential distribution, which is given as

$$F_p(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ 1 - \left(1 - \frac{x}{p+1}\right)^p, & \text{if } x \in (0, p+1) \\ 1, & \text{if } x \geq p+1, \end{cases}$$

where $p > 0$ (see Sinner *et al.* [18]).

Table 3 shows the maximum absolute relative errors of the approximations for various values of d and Δ ($x \in (-\Delta, \Delta)$), where $\lambda = 1$. Figure 2 shows the plots of the GEF, exponential function and their absolute relative difference for $\lambda = 1$, $d = 100$, $g(x) = x$.

Example 3. Let $g_\alpha(x) = \left(\frac{1-x}{x}\right)^\alpha$, $x \in [0, 1]$ and $\alpha \neq 0$. It should be added that the function g_α is known as the additive generator of the Dombi operators in

Table 3: The maximum absolute relative errors of the approximations of $e^{\lambda x}$ using $\varepsilon_{d,g}^{(\lambda)}(x)$, for $\lambda = 1$ and $g(x) = x$.

d	$x \in (-2, 2)$	$x \in (-5, 5)$	$x \in (-10, 10)$	$x \in (-20, 20)$
100	2.00×10^{-2}	1.21×10^{-1}	4.15×10^{-1}	9.01×10^{-1}
1000	2.00×10^{-3}	1.25×10^{-2}	4.91×10^{-2}	1.83×10^{-1}

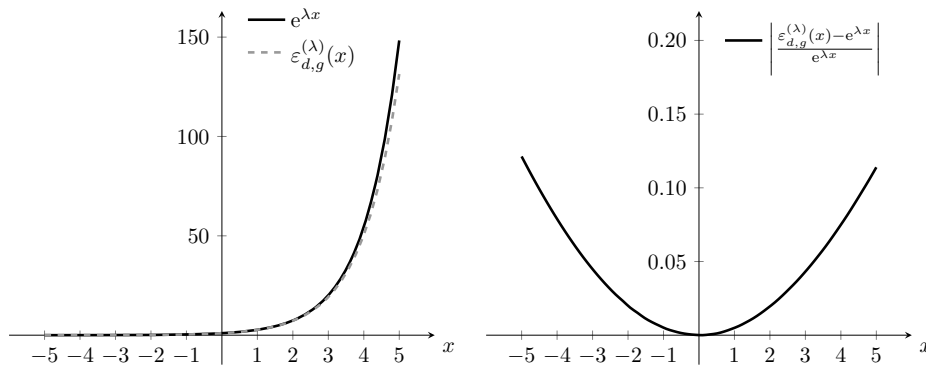


Figure 2: Plots of the GEF, exponential function and their absolute relative difference for $\lambda = 1$, $d = 100$, $g(x) = x$.

continuous-valued logic (see [7]). Clearly, $g_\alpha \in \mathcal{G}$, i.e., g_α satisfies the requirements for a generator of a GEF. Exploiting Proposition 1, we get that for any $\alpha \neq 0$, g_α induces the same GEF independently of the value of α . Let $\alpha = 1$, and let $g(x) = g_\alpha(x) = \frac{1-x}{x}$, $x \in [0, 1]$. Then, $g(\frac{1}{2}) = 1$, $g'(\frac{1}{2}) = -4$ and via direct calculation, we get that the GEF induced by g is

$$\varepsilon_{d,g}^{(\lambda)}(x) = \left(\frac{d+x}{d-x} \right)^{\lambda \frac{d}{2}}, \tag{12}$$

where $d > 0$ and $x \in [-d, d]$. Note that this generalized epsilon function is identical to the epsilon function given in Definition 1, which was introduced in [9]. Table 4 shows the maximum absolute relative errors of the approximations for various values of d and Δ ($x \in (-\Delta, \Delta)$), where $\lambda = 1$.

Figure 3 shows the plots of the GEF, exponential function and their absolute relative difference for $\lambda = 1$, $d = 100$, $g(x) = \frac{1-x}{x}$.

Notice that in Example 2, $g(0) = 0$ and $g(1) = 1$, i.e., $g(1)$ is finite, while in Example 3, $g(0) = \infty$ (more precisely, $\lim_{x \rightarrow 0} g(x) = \infty$) and $g(1) = 0$. We can see that in this latter case, we obtained a much lower maximum absolute relative approximation error. It is worth noting that based on Proposition 1, the generator $g(x) = \frac{x}{1-x}$ induces the same GEF as that in Eq. (12).

Table 4: The maximum absolute relative errors of the approximations of $e^{\lambda x}$ using $\varepsilon_{d,g}^{(\lambda)}(x)$, for $\lambda = 1$ and $g(x) = \frac{1-x}{x}$.

d	$x \in (-2, 2)$	$x \in (-5, 5)$	$x \in (-10, 10)$	$x \in (-20, 20)$
100	2.67×10^{-4}	4.18×10^{-3}	3.41×10^{-2}	3.14×10^{-1}
1000	2.67×10^{-6}	4.17×10^{-5}	3.33×10^{-4}	2.67×10^{-3}

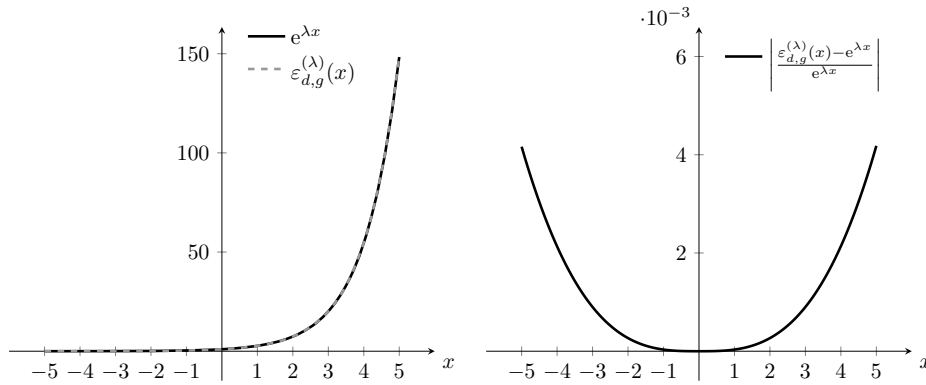


Figure 3: Plots of the GEF, exponential function and their absolute relative difference for $\lambda = 1$, $d = 100$, $g(x) = \frac{1-x}{x}$.

Remark 2. It should be added that if the generator of a GEF is a rational function, then the GEF is rational as well (see, e.g., Eq. (12)). In such a case, the generalized epsilon function is computationally simple, it may be treated as an easy-to-compute alternative to the exponential function.

4 Some applications of the generalized epsilon function

Since the exponential function may be viewed as an asymptotic generalized epsilon function, this latter may have a considerable application potential in many areas of science. Here, we will briefly present two particular applications: the first one is an approximation to the exponential distribution, the second one is an approximation to the sigmoid function.

4.1 An approximation to the exponential probability distribution

The exponential probability distribution plays an important role in probability theory and mathematical statistics (see, e.g., [6, 5, 2, 3]). Now, we will demonstrate how the cumulative distribution function (CDF) of the random variable, which has an exponential probability distribution with a $\lambda > 0$ parameter value, can be approximated using the generalized epsilon function.

Proposition 2. *Let $g \in \mathcal{G}$ such that g is either strictly increasing with $g(0) = 0$ and $g(1) = \infty$, or it is strictly decreasing with $g(1) = 0$ and $g(0) = \infty$. Furthermore, let $\lambda > 0$, $d > 0$ and let $\varepsilon_{d,g}^{(\lambda)}: [-d, d] \rightarrow \overline{\mathbb{R}}_+$ be a GEF induced by g according to Eq. (7). Then the function $F_{d,g}^{(\lambda)}: \mathbb{R} \rightarrow [0, 1]$ given by*

$$F_{d,g}^{(\lambda)}(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ 1 - \varepsilon_{d,g}^{(-\lambda)}(x), & \text{if } 0 < x \leq d \\ 1, & \text{if } x > d \end{cases} \tag{13}$$

is a CDF of a continuous random variable and for any $x \in \mathbb{R}$,

$$\lim_{d \rightarrow \infty} F_{d,g}^{(\lambda)}(x) = 1 - e^{-\lambda x}. \tag{14}$$

Proof. Clearly, $F_{d,g}^{(\lambda)}(x)$ is continuous and it satisfies the requirements for a CDF, while Eq. (14) is an immediate consequence of Theorem 2. \square

Remark 3. Utilizing the generator $g_\alpha(x) = \left(\frac{1-x}{x}\right)^\alpha$, $x \in [0, 1]$ and $\alpha \neq 0$, we get

$$F_{d,g_\alpha}^{(\lambda)}(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ 1 - \left(\frac{d+x}{d-x}\right)^{-\lambda \frac{d}{2}}, & \text{if } 0 < x < d \\ 1, & \text{if } x \geq d, \end{cases}$$

which is the CDF of the epsilon probability distribution (see [9]). Therefore, the CDF given in Eq. (13) may be treated as a generator function-based generalization of the CDF of the epsilon probability distribution. It is worth noting that $F_{d,g_\alpha}^{(\lambda)}$ approximates the exponential CDF quite well even for small values of the parameter d . For example, for $\lambda = 1$ and $d = 10$, we have

$$\max_{x \in (0,10)} \left| 1 - e^{-\lambda x} - F_{d,g_\alpha}^{(\lambda)}(x) \right| < 4.53 \times 10^{-3}$$

and

$$\max_{x \in (0,10)} \left| \frac{1 - e^{-\lambda x} - F_{d,g_\alpha}^{(\lambda)}(x)}{1 - e^{-\lambda x}} \right| < 2.03 \times 10^{-3}.$$

4.2 An approximation to the sigmoid function

It is well-known that the sigmoid function $\sigma^{(\lambda)}: \mathbb{R} \rightarrow (0, 1)$, which is given by

$$\sigma^{(\lambda)}(x) = \frac{1}{1 + e^{-\lambda x}}, \quad (15)$$

where $\lambda \in \mathbb{R} \setminus \{0\}$, has a lot of applications in many areas including computer science, engineering, biology and economics (see, e.g., [11, 12, 17, 14, 16]). It should be noted that the sigmoid function is also known as the logistic function. For example, in probability theory and mathematical statistics the logistic function can be utilized as a cumulative distribution function (logistic distribution) or as a regression function (logistic regression). The following proposition is an immediate consequence of Theorem 2.

Proposition 3. *Let $g \in \mathcal{G}$ such that g is either strictly increasing with $g(0) = 0$, or it is strictly decreasing with $g(1) = 0$. Furthermore, let $\lambda \in \mathbb{R} \setminus \{0\}$, $d > 0$ and let $\varepsilon_{d,g}^{(\lambda)}: [-d, d] \rightarrow \overline{\mathbb{R}}_+$ be a GEF induced by g according to Eq. (7). Then, for any $x \in \mathbb{R}$,*

$$\lim_{d \rightarrow \infty} \frac{1}{1 + \varepsilon_{d,g}^{(-\lambda)}(x)} = \frac{1}{1 + e^{-\lambda x}}. \quad (16)$$

Exploiting the result of Proposition 3, the function $S_{d,g}^{(\lambda)}: [-d, d] \rightarrow [0, 1]$, which is given by

$$S_{d,g}^{(\lambda)}(x) = \frac{1}{1 + \varepsilon_{d,g}^{(-\lambda)}(x)}.$$

may be viewed as a viable alternative to the sigmoid function on the bounded domain $[-d, d]$.

Remark 4. More generally, if g is either strictly increasing with $g(0) = 0$ and $g(1) = \infty$, or g is strictly decreasing with $g(1) = 0$ and $g(0) = \infty$, then the function

$$\sigma_{d,g}^{(\lambda)}(x) = g^{-1} \left(\left(\frac{g\left(\frac{x+d}{2d}\right)}{g\left(\frac{1}{2}\right)} \right)^{-2\lambda d \frac{g\left(\frac{1}{2}\right)}{g'\left(\frac{1}{2}\right)}} \right),$$

may be treated as an alternative to the sigmoid function on the bounded domain $[-d, d]$. Clearly, with the choice $g(x) = \frac{1-x}{x}$, $x \in [0, 1]$, $\sigma_{d,g}^{(\lambda)}(x) = S_{d,g}^{(\lambda)}(x)$ for any $x \in [-d, d]$.

5 Conclusions

In this study, we presented the generalized epsilon function, which is a generator function-based mapping from the bounded domain $[-d, d]$ to the non-negative extended real line ($d > 0$). We proved that if $d \rightarrow \infty$, then the generalized epsilon

function coincides with the exponential function. This result allows us to treat the generalized epsilon function as an alternative to exponential function on a bounded domain. Since this new function is generator function-dependent, it is very flexible and it can be utilized in a wide range of applications.

References

- [1] Acar, Tuncer, Mursaleen, Mohammad, and Deveci, Serife Nur. Gamma operators reproducing exponential functions. *Advances in Difference Equations*, 2020(1):1–13, 2020. DOI: 10.1186/s13662-020-02880-x.
- [2] Afify, Ahmed Z and Mohamed, Osama Abdo. A new three-parameter exponential distribution with variable shapes for the hazard rate: Estimation and applications. *Mathematics*, 8(1):135, 2020. DOI: 10.3390/math8010135.
- [3] Albanbay, Nurtay, Medetov, Bekbolat, and Zaks, Michael A. Exponential distribution of lifetimes for transient bursting states in coupled noisy excitable systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9):093105, 2021. DOI: 10.1063/5.0059102.
- [4] Aral, Ali, Cardenas-Morales, Daniel, and Garrancho, Pedro. Bernstein-type operators that reproduce exponential functions. *Journal of Mathematical Inequalities*, 12(3):861–872, 2018. DOI: 10.7153/jmi-2018-12-64.
- [5] Bakouch, Hassan S, Hussain, Tassaddaq, Chesneau, Christophe, and Khan, Muhammad Nauman. On a weighted exponential distribution with a logarithmic weight: Theory and applications. *Afrika Matematika*, pages 1–22, 2021. DOI: 10.1007/s13370-020-00861-7.
- [6] Balakrishnan, K. *Exponential distribution: theory, methods and applications*. Routledge, 2019.
- [7] Dombi, J. General class of fuzzy operators, the De Morgan class of fuzzy operators and fuzziness included by fuzzy operators. *Fuzzy Sets and Systems*, 8:149–168, 1982.
- [8] Dombi, József and Jónás, Tamás. Approximations to the normal probability distribution function using operators of continuous-valued logic. *Acta Cybernetica*, 23(3):829–852, 2018. DOI: 10.14232/actacyb.23.3.2018.7.
- [9] Dombi, József, Jónás, Tamás, and Tóth, Zsuzsanna Eszter. The epsilon probability distribution and its application in reliability theory. *Acta Polytechnica Hungarica*, 15(1):197–216, 2018. DOI: 10.12700/APH.15.1.2018.1.12.
- [10] Grabisch, Michel, Marichal, Jean-Luc, Mesiar, Radko, and Pap, Endre. Aggregation functions: Means. *Information Sciences*, 181(1):1–22, 2011. DOI: 10.1016/j.ins.2010.08.043.

- [11] Iliev, A, Kyurkchiev, Nikolay, and Markov, S. On the approximation of the step function by some sigmoid functions. *Mathematics and Computers in Simulation*, 133:223–234, 2017.
- [12] Khairunnahar, Laila, Hasib, Mohammad Abdul, Rezanur, Razib Hasan Bin, Islam, Mohammad Rakibul, and Hosain, Md Kamal. Classification of malignant and benign tissue with logistic regression. *Informatics in Medicine Unlocked*, 16:100189, 2019. DOI: 10.1016/j.imu.2019.100189.
- [13] Klement, E.P., Mesiar, R., and Pap, E. *Triangular Norms*. Trends in Logic. Springer Netherlands, 2013.
- [14] Liu, W., Wang, Z., Yuan, Y., Zeng, N., Hone, K., and Liu, X. A novel sigmoid-function-based adaptive weighted particle swarm optimizer. *IEEE Transactions on Cybernetics*, 51(2):1085–1093, 2021. DOI: 10.1109/TCYB.2019.2925015.
- [15] Musa, Salihu S, Zhao, Shi, Wang, Maggie H, Habib, Abdurrazaq G, Mustapha, Umar T, and He, Daihai. Estimation of exponential growth rate and basic reproduction number of the coronavirus disease 2019 (COVID-19) in Africa. *Infectious Diseases of Poverty*, 9(1):1–6, 2020. DOI: 10.1186/s40249-020-00718-y.
- [16] Qiao, Junfei, Li, Sanyi, and Li, Wenjing. Mutual information based weight initialization method for sigmoidal feedforward neural networks. *Neurocomputing*, 207:676–683, 2016. DOI: 10.1016/j.neucom.2016.05.054.
- [17] Qin, Y., Wang, X., and Zou, J. The optimized deep belief networks with improved logistic sigmoid units and their application in fault diagnosis for planetary gearboxes of wind turbines. *IEEE Transactions on Industrial Electronics*, 66(5):3814–3824, 2019. DOI: 10.1109/TIE.2018.2856205.
- [18] Sinner, Corinne, Dominicy, Yves, Ley, Christophe, Trufin, Julien, and Weber, Patrick. An interpolating family of size distributions, 2016. arXiv preprint arXiv:1606.04430.
- [19] Yerokhin, Vladimir A, Patkóš, Vojtěch, and Pachucki, Krzysztof. Atomic structure calculations of helium with correlated exponential functions. *Symmetry*, 13(7):1246, 2021. DOI: 10.3390/sym13071246.

Received: 16th November 2021

Dense Languages and Non Primitive Words

Toshihiro Koga^{ab}

Abstract

In this paper, we are concerned with dense languages and non primitive words. A language L is said to be dense if any string can be found as a substring of element of L . It is known that if a regular language R is dense, then R contains infinitely many non-primitive words. Then it is natural to ask whether this result can be generalized for a wider class of dense languages. In this paper, we actually obtain such generalization.

Keywords: dense languages, primitive words, monoid

1 Introduction

1.1 Density and asymptotic density

Let Σ be a non-empty finite set of distinct symbols with $|\Sigma| \geq 2$. A language $L \subseteq \Sigma^*$ is said to be dense (in Σ^*) iff $\Sigma^*s\Sigma^* \cap L \neq \emptyset$ for any $s \in \Sigma^*$, and L is said to be thin (in Σ^*) iff L is not dense in Σ^* . The concept of dense languages is important in code theory (e.g., [1, 2]), and many classifications and properties of dense languages are already known (e.g., [6, 8, 13]). Next, for $L \subseteq \Sigma^*$ and $n \geq 0$, let $D_n(L) := |L \cap \Sigma^n|/|\Sigma^n|$. Moreover, let $D^*(L) := \lim_n (1/n) \sum_{i=0}^{n-1} D_i(L)$ (asymptotic density of L), provided that the limit exists. Then L is said to have positive asymptotic density iff $D^*(L)$ exists and $D^*(L) > 0$. Although $D^*(L)$ does not necessarily exist in general, we can easily show (by Theorem III.6.1 of [12]) that if R is a regular language, then $D^*(R)$ always exists. Moreover, the same Theorem III.6.1 implies that if R is regular, then $D^*(R) = 0$ iff $\lim_n D_n(R) = 0$. Some other basic properties of asymptotic density can be found in Chapter 13 of [2].

1.2 Dömösi-Horváth-Ito conjecture

Let **REG** be the family of all regular languages over Σ and **CFL** be the family of all context-free languages over Σ . Let $Q_\Sigma \subseteq \Sigma^+$ be the set of all primitive words. In formal language theory, Dömösi-Horváth-Ito conjecture states that $Q_\Sigma \notin \mathbf{CFL}$.

^a#D-804 Purimasitei 4-1-1, Nagatsutaminamidai, Midori-ku, Yokohama-shi, Kanagawa-ken 226-0018, Japan

^bE-mail: toshihiro1123_f_ma_mgkvv@w7.dion.ne.jp, ORCID: 0000-0001-6016-1306

This conjecture was first suggested in [3], and still remains open. Some strategies for approaching this conjecture can be found in [4]. In 2020, Ryoma Syn'ya [16] suggested a new strategy for approaching $Q_\Sigma \notin \mathbf{CFL}$. He proved that any regular language with positive asymptotic density always contains infinitely many non-primitive words. Precisely, his theorem can be stated as follows:

Theorem 1.1 (Ryoma Sin'ya [16]). *Let $R \in \mathbf{REG}$ satisfy $D^*(R) > 0$. Then there exists $z \in \Sigma^+$ and $p \geq 1$ such that $z^{pn+1} \in R$ ($\forall n \geq 0$). In particular, we cannot have $R \subseteq Q_\Sigma$ for such R .*

This result states that Q_Σ does not have good lower approximations by regular languages. Since $D^*(Q_\Sigma) = 1$, we obtain $Q_\Sigma \notin \mathbf{CFL}$, provide that we have:

Claim 1.1. *Let $L \in \mathbf{CFL}$ satisfy $D^*(L) = 1$. Then there exists $R \in \mathbf{REG}$ such that $R \subseteq L$ and $D^*(R) > 0$.*

If this claim is true, then in view of $D^*(Q_\Sigma) = 1$, we can conclude from Theorem 1.1 and Claim 1.1 that $Q_\Sigma \notin \mathbf{CFL}$. However, in fact, the above Claim 1.1 is actually false. A counter-example is implicitly shown in [16, Theorem 14], and directly shown in [17]. Specifically, let $\Sigma = \{a, b\}$ and $L = \{v \in \Sigma^* \mid |v|_a \leq 2|v|_b\}$. Then we can show that this is a counter-example. Therefore, we need some generalizations of Theorem 1.1 if we continue his strategy. Aside from this, Theorem 1.1 itself is of independent interest, because this result states a non-trivial connection between asymptotic density and primitive words. In this paper, we are concerned with such connections, and we generalize Theorem 1.1 for a wider class of dense languages.

2 Main result

In this section, we state our main result. We first begin with a connection between density and positive asymptotic density for regular languages:

Theorem 2.1 (Ryoma Sin'ya [15]). *Let $R \in \mathbf{REG}$. Then $\lim_n D_n(R) = 0$ if and only if R is thin.*

A simple proof of this theorem can also be found in [7]. As we have already mentioned in Section 1, if R is regular, then $\lim_n D_n(R) = 0$ iff $D^*(R) = 0$. Moreover, if R is regular, then $D^*(R)$ always exists. Combining these with Theorem 2.1, it follows that if R is regular, then R is dense iff R has positive asymptotic density. Hence, we can restate Theorem 1.1 as follows:

Theorem 2.2. *Let $R \in \mathbf{REG}$ be dense. Then there exists $z \in \Sigma^+$ and $p \geq 1$ such that $z^{pn+1} \in R$ ($\forall n \geq 0$).*

As we have just mentioned, Theorem 2.2 is equivalent to Theorem 1.1, Now we generalize Theorem 2.2 for a wider class of dense languages. We first introduce some notations. Let $\mathbf{TL} := \{L \subseteq \Sigma^* \mid L \text{ is thin}\}$, i.e., \mathbf{TL} is the set of all thin languages over Σ . Next, For any set X , let 2^X denote the power set of X . For any

$\mathcal{N} \subseteq 2^{\Sigma^*}$, we define $\Gamma(\mathcal{N}) \subseteq 2^{\Sigma^*}$ as the regular closure of \mathcal{N} . In other words, we define $\Gamma(\mathcal{N})$ as the smallest set such that

$$\mathcal{N} \subseteq \Gamma(\mathcal{N}), \forall L_1, L_2 \in \Gamma(\mathcal{N}) [L_1 \cup L_2, L_1L_2, L_1^* \in \Gamma(\mathcal{N})].$$

Then, our result can be stated as follows:

Main Theorem 2.3. *Let $L \in \Gamma(\mathbf{TL})$ be dense. Then we have*

$$\forall u, v \in \Sigma^*, \exists z \in \Sigma^*v\Sigma^*, \exists p \geq 1, \forall n \geq 0 [(zu)^{pn}z \in L]. \tag{1}$$

Since $\mathbf{REG} = \Gamma(\{\emptyset\} \cup \{a \mid a \in \Sigma\})$ and $\{\emptyset\} \cup \{a \mid a \in \Sigma\} \subseteq \mathbf{TL}$, we have $\mathbf{REG} \subseteq \Gamma(\mathbf{TL})$, and in fact $\mathbf{REG} \subsetneq \Gamma(\mathbf{TL})$. Moreover, if $L \subseteq \Sigma^*$ satisfies the condition (1), then there exists $z \in \Sigma^+$ and $p \geq 1$ such that $z^{pn+1} \in L$ ($\forall n \geq 0$). This implies that Theorem 2.2 is just a special case of Main Theorem 2.3. In other words, Main Theorem 2.3 is a generalization of Theorem 2.2 (and Theorem 1.1).

The rest of this paper is structured as follows. In Section 3, we provide some lemmas related to monoids. In Section 4, we prove Main Theorem 2.3. In Section 5, we show that Main Theorem 2.3 is a non-trivial generalization of Theorem 2.2. In Section 6, we state some remarks. In Section 7, we state related work.

We assume that the reader is familiar with Regular languages and semigroup theory. For basic information about these topics, see, e.g., [10].

3 Some lemmas related to monoids

In this section, we provide some lemmas related to monoids.

Definition 3.1. *Let X be a monoid. Then $L \subseteq X$ is said to be dense in X iff $XsX \cap L \neq \emptyset$ for any $s \in X$, and L is said to be thin in X iff L is not dense in X .*

Lemma 3.1. *Let X be a monoid. Let $n \geq 1$ and $A_1, \dots, A_n \subseteq X$. If $\cup_{i=1}^n A_i$ is dense in X , then A_i is dense in X for some $i \in [1, n]$.*

Proof. The proof is essentially the same as [4, Proposition 2.2.1]. □

Lemma 3.2. *Let $A_1, A_2 \subseteq \Sigma^*$. If A_1A_2 is dense in Σ^* , then A_i is dense in Σ^* for some $i \in \{1, 2\}$.*

Proof. Suppose that A_1 and A_2 are thin. Then $\Sigma^*v_i\Sigma^* \cap A_i = \emptyset$ for some $v_i \in \Sigma^*$ ($i = 1, 2$). Since A_1A_2 is dense, we have $\Sigma^*v_1v_2\Sigma^* \cap A_1A_2 \neq \emptyset$, so there exists $x, y \in \Sigma^*$ and $a_i \in A_i$ ($i = 1, 2$) such that $xv_1v_2y = a_1a_2$. Then, v_1 is a substring of a_1 , or v_2 is a substring of a_2 . This contradicts the definition of v_1 and v_2 . □

Lemma 3.3. *Let M be a finite monoid. Then we have the following:*

- (i) *Let $t, x, y \in M$ satisfy $t = xty$. Then $x^mt = t = ty^m$ for some $m \geq 1$.*
- (ii) *Let $t, u, x, y \in M$ satisfy $t = xtuty$. Then $(tu)^pt = t$ for some $p \geq 1$. In particular, for any $n \geq 1$ we have $(tu)^{pn}t = t$.*

Proof. (i): Since M is finite, we have $\exists m \geq 1, \forall z \in M$ [z^m is idempotent] (see [10, Proposition 6.33]). Now assume that $t = xty$. Then $t = xty = x^2ty^2 = \dots = x^mty^m$, so $x^mt = (x^m)^2ty^m = x^mty^m = t$. Similarly, $ty^m = t$.

(ii): If $t = xtuty$, then $t = (x)t(uty)$, so $t = t(uty)^m$ for some $m \geq 1$ by (i). Then $t = t(uty)(uty)^{m-1} = (tu)t(y(uty)^{m-1})$, so $(tu)^pt = t$ for some $p \geq 1$ by (i). \square

Lemma 3.4. *Let M be a finite monoid. Let X be a monoid. Let $\eta : X \rightarrow M$ be a monoid homomorphism. Let $S \subseteq M$. Let $R := \eta^{-1}(S)$ ($\subseteq X$). If R is dense in X , then we have the following:*

$$\forall u, v \in X, \exists z \in XvX, \exists p \geq 1, \forall n \geq 0 [(zu)^{pn}z \in R].$$

Proof. Since R is dense in X , we have $R \neq \emptyset$. In view of $R = \eta^{-1}(S)$, we have $S \neq \emptyset$. Next, we have $R = \eta^{-1}(S) = \cup_{t \in S} \eta^{-1}(\{t\})$. Since R is dense, $\cup_{t \in S} \eta^{-1}(\{t\})$ is also dense. Since “ $\cup_{t \in S}$ ” is a non-empty finite union, we can apply Lemma 3.1, so $\eta^{-1}(\{t\})$ is dense for some $t \in S$. Now let $u, v \in X$ be arbitrary. Since $\eta^{-1}(\{t\})$ is dense in X , we have $XvX \cap \eta^{-1}(\{t\}) \neq \emptyset$, so $xvy \in \eta^{-1}(\{t\})$ for some $x, y \in X$. Let $z := xvy$. Then $z \in XvX$ and $\eta(z) = t$. Next, since $\eta^{-1}(\{t\})$ is dense, we have $XzuzX \cap \eta^{-1}(\{t\}) \neq \emptyset$, so $x'zuz'y' \in \eta^{-1}(\{t\})$ for some $x', y' \in X$. Then $\eta(x'zuz'y') = t$, i.e., $\eta(x')\eta(z)\eta(u)\eta(z)\eta(y') = t$. Keeping in mind $\eta(z) = t$, we have $\eta(x')t\eta(u)t\eta(y') = t$. By assumption on M , we can apply (ii) of Lemma 3.3, so there exists $p \geq 1$ such that $(t\eta(u))^{pn}t = t$ ($\forall n \geq 1$). Since $\eta(z) = t$, we have $\eta((zu)^{pn}z) = (t\eta(u))^{pn}t = t$ ($\forall n \geq 1$), so $(zu)^{pn}z \in \eta^{-1}(\{t\}) \subseteq R$ ($\forall n \geq 1$). In addition, if $n = 0$, then $(zu)^{pn}z = z \in \eta^{-1}(\{t\}) \subseteq R$. In summary,

$$z \in XvX, p \geq 1, (zu)^{pn}z \in R (\forall n \geq 0).$$

Thus we complete the proof. \square

Lemma 3.5. *Let X be a monoid. Let $X_0 \subseteq X$ be a submonoid. Let Q be a non-empty finite set. Let $L \subseteq X$ and $s \in X$. Let $R : (Q \times Q) \rightarrow 2^X$. Assume that*

$$(i) \forall n \geq 1, \forall x_1, \dots, x_n \in X_0, \exists p_0, \dots, p_n \in Q, \forall i \in [1, n] [x_i \in R(p_{i-1}, p_i)],$$

$$(ii) \forall p, q \in Q, \exists t_0, t_1 \in X_0 [t_0sR(p, q)st_1 \subseteq L],$$

$$(iii) \forall p, q, r \in Q [R(p, q)sR(q, r) \subseteq R(p, r)].$$

Then we have the following:

$$\forall x, y \in X_0, \exists z \in XyX, \exists p \geq 1, \forall n \geq 0 [(zx)^{pn}z \in L]. \quad (2)$$

Proof. STEP1: Let c_x ($\forall x \in X_0$) be new distinct symbols, and let $\Sigma_0 := \{c_x \mid x \in X_0\}$. Note that Σ_0 can be an infinite set (of distinct symbols). We can trivially verify

$$\forall n \geq 1, \forall v = v_1v_2 \cdots v_n \in \Sigma_0^n, \exists x_1, \dots, x_n \in X_0, \forall i \in [1, n] [v_i = c_{x_i}].$$

Next, let M be the set of all maps from 2^Q to 2^Q . For any $f, g \in M$, we define $f \circ g \in M$ as $(f \circ g)(U) := g(f(U))$ ($\forall U \in 2^Q$). We also define $id_{2^Q} \in M$ as

$id_{2^Q}(U) := U$ ($\forall U \in 2^Q$). Note that (M, \circ, id_{2^Q}) is a finite monoid. We define a monoid homomorphism $\eta : \Sigma_0^* \rightarrow M$ as follows: Let $\varepsilon \in \Sigma_0^*$ be the empty string. We first define $\eta(\varepsilon) := id_{2^Q}$. Next, for $x \in X_0$, we define $\eta(c_x) \in M$ as

$$\eta(c_x)(U) := \{q \in Q \mid \exists p \in U [x \in R(p, q)]\} \text{ for } U \in 2^Q. \quad (3)$$

Next, for $n \geq 2$ and $v = v_1 v_2 \cdots v_n \in \Sigma_0^n$, we define $\eta(v) := \eta(v_1) \circ \eta(v_2) \circ \cdots \circ \eta(v_n)$. By this definition, we can easily show that $\eta : \Sigma_0^* \rightarrow M$ is a monoid homomorphism. Moreover, by induction on $|v| \geq 0$, we can easily verify the following:

$$\forall v \in \Sigma_0^*, \forall U_1, U_2 \in 2^Q [U_1 \subseteq U_2 \Rightarrow \eta(v)(U_1) \subseteq \eta(v)(U_2)]. \quad (4)$$

STEP2: For any $p, q \in Q$, we define $A_{p,q} := \{v \in \Sigma_0^* \mid q \in \eta(v)(\{p\})\}$. Note that $\varepsilon \in A_{p,p}$ for any $p \in Q$. Moreover, we can trivially verify that

$$\forall v, w \in \Sigma_0^* [\eta(v) = \eta(w) \Rightarrow \forall p, q \in Q [v \in A_{p,q} \Leftrightarrow w \in A_{p,q}]]. \quad (5)$$

Next, we show

$$\forall p, q, r \in Q [A_{p,q} A_{q,r} \subseteq A_{p,r}]. \quad (6)$$

Let $v \in A_{p,q}$ and $w \in A_{q,r}$ be arbitrary. Then $q \in \eta(v)(\{p\})$ and $r \in \eta(w)(\{q\})$. In particular, $\{q\} \subseteq \eta(v)(\{p\})$. By (4), we have $\eta(w)(\{q\}) \subseteq \eta(w)(\eta(v)(\{p\})) = (\eta(v) \circ \eta(w))(\{p\}) = \eta(vw)(\{p\})$. In view of $r \in \eta(w)(\{q\})$, we have $r \in \eta(vw)(\{p\})$, so $vw \in A_{p,r}$. Thus we obtain (6). Next, we show

$$\forall v \in \Sigma_0^*, \exists p, q \in Q [v \in A_{p,q}]. \quad (7)$$

Since $\varepsilon \in A_{p,p}$ for any $p \in Q$, we have only to show

$$\forall v \in \Sigma_0^+, \exists p, q \in Q [v \in A_{p,q}].$$

Let $n \geq 1$ and $v = v_1 \cdots v_n \in \Sigma_0^n$ be arbitrary. There exists $x_1, \dots, x_n \in X_0$ such that $v_i = c_{x_i}$ ($\forall i \in [1, n]$). By (i), there exists $p_0, \dots, p_n \in Q$ such that $x_i \in R(p_{i-1}, p_i)$ ($\forall i \in [1, n]$). For any $i \in [1, n]$, it follows from (3) that

$$\begin{aligned} \eta(c_{x_i})(\{p_{i-1}\}) &= \{q \in Q \mid \exists p \in \{p_{i-1}\} [x_i \in R(p, q)]\} \\ &= \{q \in Q \mid x_i \in R(p_{i-1}, q)\} \ni p_i, \end{aligned}$$

i.e., $p_i \in \eta(c_{x_i})(\{p_{i-1}\})$, so $c_{x_i} \in A_{p_{i-1}, p_i}$. In view of (6), we have

$$v = c_{x_1} c_{x_2} \cdots c_{x_n} \in A_{p_0, p_1} A_{p_1, p_2} \cdots A_{p_{n-1}, p_n} \subseteq A_{p_0, p_n}.$$

Thus we obtain (7).

STEP3: We define $g : \Sigma_0^+ \rightarrow X$ as follows: For any $x \in X_0$, we define $g(c_x) := x$. For any $n \geq 2$ and $v = v_1 v_2 \cdots v_n \in \Sigma_0^n$, we define $g(v) := g(v_1) s g(v_2) s \cdots s g(v_n)$. Note that we have $g(vw) = g(v) s g(w)$ for any $v, w \in \Sigma_0^+$. Then we can easily verify

$$\forall v, w \in \Sigma_0^+, \forall n \geq 1 [g(v^n w) = (g(v) s)^n g(w)]. \quad (8)$$

Next, we show

$$\forall v \in \Sigma_0^+, \forall p, q \in Q [v \in A_{p,q} \Rightarrow g(v) \in R(p, q)]. \quad (9)$$

The proof is by induction on $|v| \geq 1$. We first show the case $|v| = 1$. Let $v \in \Sigma_0^1$ and $p, q \in Q$ satisfy $v \in A_{p,q}$. We have $v = c_x$ for some $x \in X_0$. In view of $v \in A_{p,q}$, we have $q \in \eta(v)(\{p\})$. In addition,

$$\begin{aligned} \eta(v)(\{p\}) &= \eta(c_x)(\{p\}) = \{q' \in Q \mid \exists p' \in \{p\} [x \in R(p', q')] \} \\ &= \{q' \in Q \mid x \in R(p, q')\}, \end{aligned}$$

so $q \in \{q' \in Q \mid x \in R(p, q')\}$, i.e., $x \in R(p, q)$. Since $g(v) = g(c_x) = x$, we obtain $g(v) \in R(p, q)$. Thus we obtain (9) for $|v| = 1$. Next, let $n \geq 1$ be arbitrary. Assume that (9) holds for $|v| = n$. Let $v \in \Sigma_0^{n+1}$ and $p, q \in Q$ satisfy $v \in A_{p,q}$. We can write $v = wc_x$ for some $w \in \Sigma_0^n$ and $x \in X_0$. In view of $v \in A_{p,q}$, we have

$$\begin{aligned} q \in \eta(v)(\{p\}) &= \eta(wc_x)(\{p\}) = (\eta(w) \circ \eta(c_x))(\{p\}) = \eta(c_x)(\eta(w)(\{p\})) \\ &= \{q' \in Q \mid \exists p' \in \eta(w)(\{p\}) [x \in R(p', q')] \}, \end{aligned}$$

so there exists $p' \in \eta(w)(\{p\})$ such that $x \in R(p', q)$. In view of $p' \in \eta(w)(\{p\})$, we have $w \in A_{p,p'}$. By inductive hypothesis, we have $g(w) \in R(p, p')$. Then $g(v) = g(wc_x) = g(w)sg(c_x) = g(w)sx \in R(p, p')sR(p', q)$. By (iii), we have $R(p, p')sR(p', q) \subseteq R(p, q)$, so $g(v) \in R(p, q)$. Thus we obtain (9) for $|v| = n + 1$. By induction, we obtain (9).

STEP4: Let $F := \eta(\Sigma_0^*) \subseteq M$. Then F is a non-empty finite set. Moreover, we trivially have $\eta : \Sigma_0^* \rightarrow F$, so $\Sigma_0^* = \cup_{f \in F} \eta^{-1}(\{f\})$. In particular, $\cup_{f \in F} \eta^{-1}(\{f\})$ is dense in Σ_0^* . By Lemma 3.1, $\eta^{-1}(\{f\})$ is dense in Σ_0^* for some $f \in F$. At this point, we obtain the following:

- M is a finite monoid, Σ_0^* is a monoid, $\eta : \Sigma_0^* \rightarrow M$ is a monoid homomorphism, $\{f\} \subseteq M$, and $\eta^{-1}(\{f\}) \subseteq \Sigma_0^*$ is dense in Σ_0^* .

Then we can apply Lemma 3.4, and we obtain

$$\forall u, v \in \Sigma_0^*, \exists z \in \Sigma_0^* v \Sigma_0^*, \exists p \geq 1, \forall n \geq 0 [(zu)^{pn} z \in \eta^{-1}(\{f\})].$$

Since $f \in F = \eta(\Sigma_0^*)$, we have $f = \eta(w)$ for some $w \in \Sigma_0^*$. Then, for any $w' \in \eta^{-1}(\{f\})$, we trivially have $\eta(w') = \eta(w)$. Thus we obtain

$$\forall u, v \in \Sigma_0^*, \exists z \in \Sigma_0^* v \Sigma_0^*, \exists p \geq 1, \forall n \geq 0 [\eta((zu)^{pn} z) = \eta(w)]. \quad (10)$$

Next, by (7), we have $w \in A_{p,q}$ for some $p, q \in Q$. By (ii), we have $t_0 s R(p, q) s t_1 \subseteq L$ for some $t_0, t_1 \in X_0$. Now let $x', y' \in X_0$ be arbitrary. Let $x := t_1 x' t_0$. Since $x', t_0, t_1 \in X_0$ and X_0 is a monoid, we have $x \in X_0$. Then $c_x, c_{y'} \in \Sigma_0 \subseteq \Sigma_0^*$, so we can apply (10), i.e., there exists $z \in \Sigma_0^* c_{y'} \Sigma_0^*$ and $p \geq 1$ such that $\eta((zc_x)^{pn} z) = \eta(w)$ ($\forall n \geq 0$). Since $w \in A_{p,q}$, we obtain $(zc_x)^{pn} z \in A_{p,q}$ ($\forall n \geq 0$) by (5). Since $z \in \Sigma_0^* c_{y'} \Sigma_0^* \subseteq \Sigma_0^+$, we have $(zc_x)^{pn} z \in \Sigma_0^+$ ($\forall n \geq 0$), so $g((zc_x)^{pn} z) \in$

$R(p, q)$ ($\forall n \geq 0$) by (9). Then $t_0sg((zc_x)^{pn}z)st_1 \in t_0sR(p, q)st_1 \subseteq L$ ($\forall n \geq 0$). In short,

$$\forall n \geq 0 [t_0sg((zc_x)^{pn}z)st_1 \in L]. \tag{11}$$

Let $z' := t_0sg(z)st_1$. We show $t_0sg((zc_x)^{pn}z)st_1 = (z'x')^{pn}z'$ ($\forall n \geq 0$). If $n = 0$, then $t_0sg((zc_x)^{pn}z)st_1 = t_0sg(z)st_1 = z' = (z'x')^{pn}z'$. If $n \geq 1$, then keeping in mind $x = t_1x't_0$ and (8), we have

$$\begin{aligned} g((zc_x)^{pn}z) &= (g(zc_x)s)^{pn}g(z) = (g(z)sg(c_x)s)^{pn}g(z) \\ &= (g(z)sx s)^{pn}g(z) = (g(z)st_1x't_0s)^{pn}g(z), \end{aligned}$$

so

$$\begin{aligned} t_0sg((zc_x)^{pn}z)st_1 &= t_0s(g(z)st_1x't_0s)^{pn}g(z)st_1 \\ &= (t_0sg(z)st_1x')^{pn}t_0sg(z)st_1 = (z'x')^{pn}z'. \end{aligned}$$

Thus we obtain $t_0sg((zc_x)^{pn}z)st_1 = (z'x')^{pn}z'$ ($\forall n \geq 0$). Combining this with (11), we obtain $(z'x')^{pn}z' \in L$ ($\forall n \geq 0$). Moreover, since $z \in \Sigma_0^*c_y'\Sigma_0^*$, we can write $z = \alpha c_y'\beta$ for some $\alpha, \beta \in \Sigma_0^*$. If $\alpha, \beta \in \Sigma_0^+$, then $g(z) = g(\alpha)sy'sg(\beta) \in Xy'X$. Similarly, we obtain $g(z) \in Xy'X$ in the case of $\alpha = \varepsilon$ or $\beta = \varepsilon$. Then $z' = t_0sg(z)st_1 \in Xy'X$. In summary, we obtain

$$\forall x', y' \in X_0, \exists z' \in Xy'X, \exists p \geq 1, \forall n \geq 0 [(z'x')^{pn}z' \in L].$$

Thus we complete the proof. □

4 Proof of Main Theorem 2.3

In this section, we prove Main Theorem 2.3. For $L \subseteq \Sigma^*$, we define $H_1(L)$ as

$$H_1(L) : \forall u, v \in \Sigma^*, \exists z \in \Sigma^*v\Sigma^*, \exists p \geq 1, \forall n \geq 0 [(zu)^{pn}z \in L].$$

Note that $H_1(L)$ is exactly the same statement as (1). Next, we define

$$\mathcal{M}_1 := \mathbf{TL} \cup \{L \subseteq \Sigma^* \mid H_1(L)\}.$$

As for this \mathcal{M}_1 , we can show the following Lemma:

Lemma 4.1. \mathcal{M}_1 is closed under regular operations, i.e., we have $L_1 \cup L_2, L_1L_2, L_1^* \in \mathcal{M}_1$ for any $L_1, L_2 \in \mathcal{M}_1$.

Once we have obtained this lemma, we can show Main Theorem 2.3 as follows:

Proof. We prove Main Theorem 2.3, provided that Lemma 4.1 is already proved. First, we trivially obtain $\mathbf{TL} \subseteq \mathcal{M}_1$. Moreover, \mathcal{M}_1 is closed under regular operations by Lemma 4.1. By the minimality of $\Gamma(\mathbf{TL})$, we obtain $\Gamma(\mathbf{TL}) \subseteq \mathcal{M}_1$. Now let $L \in \Gamma(\mathbf{TL})$ be dense. Since $\Gamma(\mathbf{TL}) \subseteq \mathcal{M}_1$, we have $L \in \mathcal{M}_1$. Then $L \in \mathbf{TL}$ or $H_1(L)$. Since L is dense, we must have $H_1(L)$, i.e., L satisfies (1). □

At this point, we have only to show Lemma 4.1. Therefore, the rest of this section is devoted to showing Lemma 4.1. First, one can trivially verify the closure of \mathcal{M}_1 under union by applying Lemma 3.1 and the following basic fact:

$$\forall A, B \subseteq \Sigma^* [[A \subseteq B, H_1(A)] \Rightarrow H_1(B)]. \quad (12)$$

Next, we show the closure under concatenation:

Proof. We first show the following:

$$\forall L_1, L_2 \subseteq \Sigma^* [[H_1(L_1) \vee H_1(L_2)] \Rightarrow [L_1L_2 = \emptyset \vee H_1(L_1L_2)]]. \quad (13)$$

Let $L_1, L_2 \subseteq \Sigma^*$ satisfy $H_1(L_1) \vee H_1(L_2)$. If $L_1L_2 = \emptyset$, then we obtain (13). Now we may assume $L_1L_2 \neq \emptyset$. Then $L_1 \neq \emptyset$ and $L_2 \neq \emptyset$, so we can take $l_1 \in L_1$ and $l_2 \in L_2$. If $H_1(L_1)$ holds, then let $u, v \in \Sigma^*$ be arbitrary. We apply $H_1(L_1)$ with l_2u and v . Then there exists $z \in \Sigma^*v\Sigma^*$ and $p \geq 1$ such that $(z(l_2u))^{pn}z \in L_1$ ($\forall n \geq 0$). Let $z' := zl_2$. Then $z' \in \Sigma^*v\Sigma^*$. Moreover, $(z'u)^{pn}z' = (zl_2u)^{pn}zl_2 = ((zl_2u)^{pn}z)l_2 \in L_1l_2 \subseteq L_1L_2$ ($\forall n \geq 0$). Thus we obtain $H_1(L_1L_2)$. Next, if $H_1(L_2)$ holds, then let $u, v \in \Sigma^*$ be arbitrary. We apply $H_1(L_2)$ with ul_1 and v . Then there exists $z \in \Sigma^*v\Sigma^*$ and $p \geq 1$ such that $(z(ul_1))^{pn}z \in L_2$ ($\forall n \geq 0$). Let $z' := l_1z$. Then $z' \in \Sigma^*v\Sigma^*$. In general, we have $(xy)^nx = x(yx)^n$ for any $x, y \in \Sigma^*$ and $n \geq 0$, so $(z'u)^{pn}z' = (l_1zu)^{pn}l_1z = l_1(zul_1)^{pn}z \in l_1L_2 \subseteq L_1L_2$ ($\forall n \geq 0$). Thus we obtain $H_1(L_1L_2)$, and we complete the proof of (13). Now the closure of \mathcal{M}_1 under concatenation trivially follows from (13), Lemma 3.2, and $\emptyset \in \mathbf{TL}$. \square

Finally, we show the closure under Kleene star. For that, we need the following:

Lemma 4.2. *Let $A \subseteq \Sigma^*$. If A is thin and A^* is dense, then we have $H_1(A^*)$.*

Proof. STEP1: Let $\varepsilon \in \Sigma^*$ be the empty string. Let $A \subseteq \Sigma^*$. Assume that A is thin, A^* is dense, and $\varepsilon \notin A$. We show $H_1(A^*)$ in this case.¹ If $A = \emptyset$, then $A^* = \{\varepsilon\}$. However, since A^* is dense, this is a contradiction. Thus we obtain $A \neq \emptyset$. Next, since A is thin, we have $\Sigma^*t\Sigma^* \cap A = \emptyset$ for some $t \in \Sigma^*$. If $t = \varepsilon$, then $\Sigma^*\Sigma^* \cap A = \emptyset$, so we must have $A = \emptyset$, which is a contradiction. Thus we obtain $t \neq \varepsilon$. Since A^* is dense, we have $\Sigma^*t\Sigma^* \cap A^* \neq \emptyset$, so $t'tt'' \in A^*$ for some $t', t'' \in \Sigma^*$. Let $s := t'tt''$. Then $s \neq \varepsilon$ and $s \in A^*$. If $\Sigma^*s\Sigma^* \cap A \neq \emptyset$, then in view of $\Sigma^*s\Sigma^* \subseteq \Sigma^*t\Sigma^*$, we have $\Sigma^*t\Sigma^* \cap A \neq \emptyset$, which is a contradiction. Thus we obtain $\Sigma^*s\Sigma^* \cap A = \emptyset$. Next, let S_{pre} be the set of all prefixes of s and S_{suf} be the set of all suffixes of s . Note that we have $\varepsilon \in S_{pre}$ and $\varepsilon \in S_{suf}$. Let

$$\begin{aligned} S'_{pre} &:= \{ \beta \in S_{pre} \mid \exists w \in \Sigma^* [w\beta \in A] \}, \\ S'_{suf} &:= \{ \gamma \in S_{suf} \mid \exists w \in \Sigma^* [\gamma w \in A] \}. \end{aligned}$$

Since $A \neq \emptyset$, it is easy to verify that $\varepsilon \in S'_{pre}$ and $\varepsilon \in S'_{suf}$. Next, let

$$Q := \{ (\beta, \alpha, \gamma) \mid \beta \in S'_{pre}, \gamma \in S'_{suf}, \alpha \in A^*, \beta\alpha\gamma = s \}.$$

¹In fact, the additional assumption $\varepsilon \notin A$ is not essential, but we adopt this assumption for simplicity.

Note that Q is a finite set. In addition, since $s \in A^*$, we have $(\varepsilon, s, \varepsilon) \in Q$, so $Q \neq \emptyset$. Next, for $p = (\beta, \alpha, \gamma) \in Q$ and $q = (\beta', \alpha', \gamma') \in Q$, we define $R(p, q) := \{x \in \Sigma^* \mid \gamma x \beta' \in A^*\}$. Let $X := \Sigma^*$, $X_0 := \Sigma^*$, and $L := A^*$. We show (i), (ii), and (iii) of Lemma 3.5.

(i): Let $n \geq 1$ and $x_1, \dots, x_n \in X_0$. We have to show there exists $p_0, \dots, p_n \in Q$ such that $x_i \in R(p_{i-1}, p_i) \ (\forall i \in [1, n])$. We first deal with the case $n = 1$. Then $x_1 \in X_0$ is given, and we have to show there exists $p_0, p_1 \in Q$ such that $x_1 \in R(p_0, p_1)$. First, since A^* is dense, we have $\Sigma^* s x_1 s \Sigma^* \cap A^* \neq \emptyset$, so $u s x_1 s v \in A^*$ for some $u, v \in \Sigma^*$. This implies that we can decompose the whole string $u s x_1 s v$ into concatenations of strings in A . Keeping in mind $\Sigma^* s \Sigma^* \cap A = \emptyset$, the decomposition for each s (in $u s x_1 s v$) is like Fig. 1. Therefore, the decomposition for $u s x_1 s v$ is like Fig. 2, so $p_0 := (\beta, \alpha, \gamma) \in Q$ and $p_1 := (\beta', \alpha', \gamma') \in Q$ in Fig. 2 satisfies $x_1 \in R(p_0, p_1)$. See also Fig. 3.

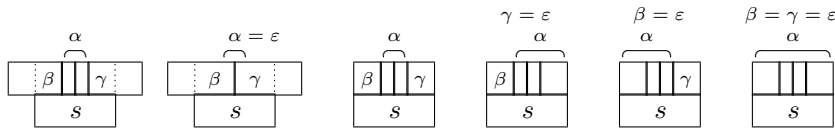


Figure 1: Six examples of decomposition for each s in $u s x_1 s v$.

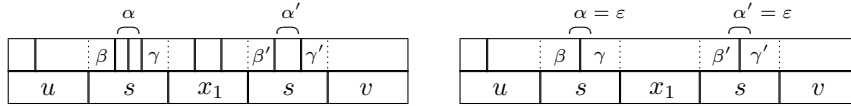


Figure 2: Two examples of decomposition for $u s x_1 s v$.

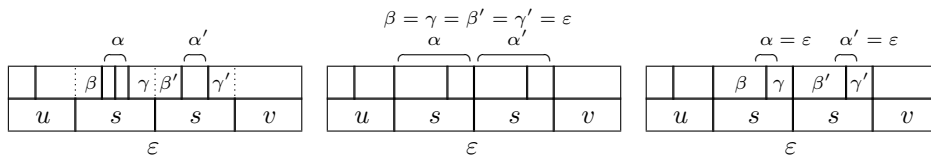


Figure 3: Three examples of decomposition for $u s x_1 s v$ with $x_1 = \varepsilon$.

In general case $n \geq 1$, we have $\Sigma^* s x_1 s \dots s x_n s \in \Sigma^* \cap A^* \neq \emptyset$, so $u s x_1 s \dots s x_n s v \in A^*$ for some $u, v \in \Sigma^*$. This implies that we can decompose the whole string $u s x_1 s \dots s x_n s v$ into concatenations of strings in A . Keeping in mind $\Sigma^* s \Sigma^* \cap A = \emptyset$, we can easily show that there exists $p_0, \dots, p_n \in Q$ such that $x_i \in R(p_{i-1}, p_i) \ (\forall i \in [1, n])$. See also Fig. 4 and Fig. 5.

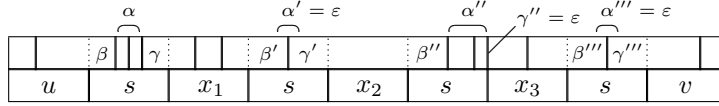


Figure 4: An example of decomposition.

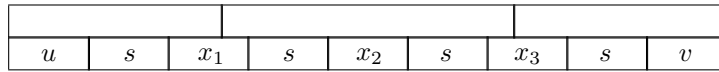


Figure 5: Decomposition like above is impossible due to $\Sigma^*s\Sigma^* \cap A = \emptyset$.

(ii): Let $p = (\beta, \alpha, \gamma) \in Q$ and $q = (\beta', \alpha', \gamma') \in Q$ be arbitrary. Since $\beta \in S'_{pre}$ and $\gamma' \in S'_{suf}$, we have $t_0\beta, \gamma't_1 \in A$ for some $t_0, t_1 \in \Sigma^*$ ($= X_0$). Let $x \in R(p, q)$ be arbitrary. Then $\gamma x \beta' \in A^*$, so $(t_0\beta)\alpha(\gamma x \beta')\alpha'(\gamma't_1) \in A^*$. Since $\beta\alpha\gamma = s$ and $\beta'\alpha'\gamma' = s$, we obtain $t_0sxt_1 \in A^*$. Since $x \in R(p, q)$ is arbitrary, we have $t_0sR(p, q)st_1 \subseteq A^*$ ($= L$), so we obtain (ii).

(iii): Let $p = (\beta, \alpha, \gamma) \in Q$, $q = (\beta', \alpha', \gamma') \in Q$, and $r = (\beta'', \alpha'', \gamma'') \in Q$ be arbitrary. Let $x \in R(p, q)$ and $y \in R(q, r)$. Then $\gamma x \beta' \in A^*$ and $\gamma' y \beta'' \in A^*$. In particular, $(\gamma x \beta')\alpha'(\gamma' y \beta'') \in A^*$. Since $\beta'\alpha'\gamma' = s$, we have $\gamma x s y \beta'' \in A^*$, so $x s y \in R(p, r)$. This implies $R(p, q)sR(q, r) \subseteq R(p, r)$. Thus we obtain (iii). Consequently, we can apply Lemma 3.5, and we obtain (2). In other words,

$$\forall x, y \in \Sigma^*, \exists z \in \Sigma^*y\Sigma^*, \exists p \geq 1, \forall n \geq 0 [(zx)^{pn}z \in A^*].$$

This implies $H_1(A^*)$.

STEP2: Let $A \subseteq \Sigma^*$. Assume that A is thin and A^* is dense. Let $B := A - \{\varepsilon\}$. In general, we have $(A - \{\varepsilon\})^* = A^*$, so $B^* = A^*$. Since A^* is dense, it follows that B^* is dense. If B is dense, then in view of $B \subseteq A$, it follows that A is dense, which is a contradiction. Therefore, B is thin. Moreover, we have $\varepsilon \notin B$. Hence, by STEP1, we have $H_1(B^*)$. Since $B^* = A^*$, we obtain $H_1(A^*)$. \square

The closure of \mathcal{M}_1 under Kleene star trivially follows from Lemma 4.2 and (12). Hence, we complete the proof of Lemma 4.1.

5 On Theorem 2.2 and Main Theorem 2.3

In this section, we prove the following theorem:

Theorem 5.1. *Let $\Sigma = \{a, b\}$. Then there exists a dense $L \in \Gamma(\mathbf{TL})$ such that there is no dense $R \in \mathbf{REG}$ with $R \subseteq L$.*

In view of this theorem, we can say that Main Theorem 2.3 is a non-trivial generalization of Theorem 2.2.

Proof. Let \mathbb{N} be the set of all positive integers. Let $I = \{(pqn)^4 + q \mid p, q, n \geq 1\} \subseteq \mathbb{N}$. We show the following:

(i) $\forall p, q \geq 1 [pm + q \in I \text{ for infinitely many } m \geq 1]$.

(ii) $\forall p, q \geq 1 [pm + q \in \mathbb{N} - I \text{ for infinitely many } m \geq 1]$.

(i): This is obvious.

(ii): For any $t \geq 1$, we can easily show $I \cap [1, t] \subseteq \{(pqn)^4 + q \mid 1 \leq p, q, n \leq t^{1/4}\}$. In particular, $|I \cap [1, t]| \leq t^{3/4}$, so $\lim_{t \rightarrow +\infty} |I \cap [1, t]|/t = 0$. Now let $p, q \geq 1$. We show $pm + q \in \mathbb{N} - I$ for infinitely many $m \geq 1$. Supposing the contrary, there exists $m_0 \geq 1$ such that $pm + q \in I$ ($\forall m \geq m_0$). Let $J := \{pm + q \mid m \geq m_0\}$, for short. Then we have $J \subseteq I$. Combining this inclusion with $\lim_{t \rightarrow +\infty} |I \cap [1, t]|/t = 0$, we have $\lim_{t \rightarrow +\infty} |J \cap [1, t]|/t = 0$. However, since $J = \{pm + q \mid m \geq m_0\}$, we have $\lim_{t \rightarrow +\infty} |J \cap [1, t]|/t = 1/p$. This is a contradiction. Thus we obtain (ii).

Next, we define $f : a\Sigma^*b \cup \{\varepsilon\} \rightarrow \mathbb{N} \cup \{0\}$ as follows: For any $v \in a\Sigma^*b$, there exists unique $k \geq 1$ and unique $n_1, m_1, \dots, n_k, m_k \geq 1$ such that $v = a^{n_1}b^{m_1} \dots a^{n_k}b^{m_k}$. Then we define $f(v) := |\{i \in [1, k] \mid n_i \in I\}|$. We also define $f(\varepsilon) := 0$. As for this f , we can easily verify the following:

$$\forall v, w \in a\Sigma^*b \cup \{\varepsilon\} [vw \in a\Sigma^*b \cup \{\varepsilon\}, f(vw) = f(v) + f(w)]. \quad (14)$$

Next, let $r \in \{0, 1\}$ be arbitrary. We define

$$L_r := \{v \in a\Sigma^*b \cup \{\varepsilon\} \mid f(v) \equiv r \pmod{2}\} \subseteq a\Sigma^*b \cup \{\varepsilon\}.$$

We show L_r satisfies the desired property. We first show that L_r is dense in Σ^* . Take an $n_1 \in I$, and let $\alpha := a^{n_1}b$. Let $v \in \Sigma^*$ be arbitrary. Then $\alpha, avb \in a\Sigma^*b$. By (14), we have $f(avb\alpha^k) = f(avb) + kf(\alpha) = f(avb) + k$ ($\forall k \geq 1$). In particular, we have $f(avb\alpha^{k_1}) \equiv r \pmod{2}$ for some $k_1 \in \{1, 2\}$. Then $avb\alpha^{k_1} \in L_r$, i.e., we have $\Sigma^*v\Sigma^* \cap L_r \neq \emptyset$. Hence, L_r is dense. Next, suppose that there exists a dense $R \in \mathbf{REG}$ such that $R \subseteq L_r$. Let G be a deterministic finite automaton which represents R . Let $t \geq 1$ be the number of all states of G . Since R is dense, we have $\Sigma^*b^2a^{t+9}ba^2\Sigma^* \cap R \neq \emptyset$. Then we have $u'b^2a^{t+9}ba^2v' \in R$ for some $u', v' \in \Sigma^*$. Let $u := u'b$ and $v := av'$. Then $u, v \in \Sigma^+$ and $uba^{t+9}bav \in R$. By the definition of $t \geq 1$, we can apply a standard pumping argument, and we can show that there exists $p, q \geq 1$ such that $uba^{pn+q}bav \in R$ ($\forall n \geq 1$). Since $R \subseteq L_r$, we have $uba^{pn+q}bav \in L_r$ ($\forall n \geq 1$). Since $L_r \subseteq a\Sigma^*b \cup \{\varepsilon\}$, we have $uba^{pn+q}bav \in a\Sigma^*b$ ($\forall n \geq 1$). Then, the first character of u must be a , and the last character of v must be b . In particular, we have $ub, av, a^{pn+q}b \in a\Sigma^*b$. By (14), we have $f(uba^{pn+q}bav) = f(ub) + f(a^{pn+q}b) + f(av)$. Since $uba^{pn+q}bav \in L_r$, we have $f(uba^{pn+q}bav) \equiv r \pmod{2}$, so $f(ub) + f(a^{pn+q}b) + f(av) \equiv r \pmod{2}$. By (i) and (ii), there exists $m, m' \geq 1$ such that $pm + q \in I$ and $pm' + q \in \mathbb{N} - I$. Then $f(ub) + 1 + f(av) \equiv r \pmod{2}$ and $f(ub) + 0 + f(av) \equiv r \pmod{2}$, which is a contradiction. Hence, there is no dense $R \in \mathbf{REG}$ with $R \subseteq L_r$. Finally, we show $L_r \in \Gamma(\mathbf{TL})$. Let

$$A := \{a^n b^m \mid n \in I, m \geq 1\}, B := \{a^n b^m \mid n \in \mathbb{N} - I, m \geq 1\}.$$

Consider the following language equations:

$$X_0 = AX_1 \cup BX_0 \cup \{\varepsilon\}, \quad X_1 = AX_0 \cup BX_1. \quad (15)$$

Let $Y_0, Y_1 \subseteq \Sigma^*$ be the least solution of (15). In fact, we can explicitly write $Y_0 = (AB^*A \cup B)^*$ and $Y_1 = B^*A(AB^*A \cup B)^*$. Since $A, B \in \mathbf{TL} \subseteq \Gamma(\mathbf{TL})$, we have $Y_0, Y_1 \in \Gamma(\mathbf{TL})$. Moreover, we can easily show that L_0, L_1 is also the least solution of (15). Hence, we must have $L_0 = Y_0$ and $L_1 = Y_1$, so $L_r \in \Gamma(\mathbf{TL})$. \square

6 Some remarks

In this section, we give some remarks.

6.1 On Lemma 4.1

Let $H_2(L)$ be a statement defined as

$$H_2(L) : \exists z \in \Sigma^+, \exists p \geq 1, \forall n \geq 0 [z^{pn+1} \in L].$$

Let $\mathcal{M}_2 := \mathbf{TL} \cup \{L \subseteq \Sigma^* \mid H_2(L)\}$. It is natural to consider \mathcal{M}_2 instead of \mathcal{M}_1 in Lemma 4.1. We would like to show that \mathcal{M}_2 is closed under regular operations. However, \mathcal{M}_2 is not closed under concatenation. For example, let $\Sigma = \{a, b\}$, $L_1 := \{va^{10|v|} \mid v \in \Sigma^+\}$, and $L_2 := \{b\}$. We can show that $L_1, L_2 \in \mathcal{M}_2$ and $L_1L_2 \notin \mathcal{M}_2$. This is why we have considered $H_1(L)$ instead of $H_2(L)$.

6.2 On Main Theorem 2.3

For any $\mathcal{L} \subseteq 2^{\Sigma^*}$, consider the following claim:

Claim 6.1. *Let $R \in \mathcal{L}$ be dense. Then there exists $z \in \Sigma^+$ and $p \geq 1$ such that $z^{pn+1} \in R$ ($\forall n \geq 0$).*

Note that Claim 6.1 with $\mathcal{L} = \mathbf{REG}$ is exactly Theorem 2.2. Moreover, Claim 6.1 with $\mathcal{L} = \Gamma(\mathbf{TL})$ is also true, as we have already shown. Keeping in mind Dömösi-Horváth-Ito conjecture, it is desirable to prove Claim 6.1 for $\mathcal{L} = \mathbf{CFL}$, because in this case we trivially obtain Dömösi-Horváth-Ito conjecture (by considering $R = Q_\Sigma$). However, in fact, Claim 6.1 does not hold even if $\mathcal{L} = \mathbf{DCFL}$ (deterministic context-free languages). Here we provide a counter-example. Let $\Sigma = \{ \langle, \rangle \}$. Let $L \subseteq \Sigma^*$ be the Dyck language over Σ . Let $R := \{v\} \mid v \in L\}$. It is easy to show that R is dense, $R \subseteq Q_\Sigma$, and $R \in \mathbf{DCFL}$. Therefore, this R is a counter-example of Claim 6.1 for $\mathcal{L} = \mathbf{DCFL}$. This fact implies that extending Theorem 2.2 is a hard problem in general. This situation is already indicated in our proofs: we have proved non-trivial lemmas to obtain Main Theorem 2.3.

7 Related work

In this section, we briefly state some related work. For $L \subseteq \Sigma^*$, let \sim_L be the syntactic equivalence of L , and let Σ^*/\sim_L be the syntactic monoid of L . By Myhill-Nerode Theorem, we can easily show that L is a regular language iff Σ^*/\sim_L is a finite monoid (see also [10, Proposition 3.18]).

7.1 Related work for dense and disjunctive language

A language $L \subseteq \Sigma^*$ is said to be disjunctive iff $\forall u, v \in \Sigma^* [u \sim_L v \Leftrightarrow u = v]$. In particular, if L is a disjunctive language, then Σ^*/\sim_L is an infinite set.

The concept of disjunctive languages is closely related to dense languages. For example, it is shown in [11, PROPOSITION 2.5] that a language L is dense iff there exists a disjunctive language L' such that $L' \subseteq L$. Many properties of disjunctive languages are already known (e.g., [11, 14]). Moreover, many connections between dense and disjunctive languages are known (e.g., [5, 6]).

7.2 Related work for Theorem 1.1, 2.2, and Main Theorem 2.3

As for Theorem 1.1, 2.2, and Main Theorem 2.3, we refer to [6, 9] as direct related works. Theorem 2.2 is exactly the same as [9, Corollary 4.6]. Next, it is shown in [6] that if $R \subseteq \Sigma^*$ is a dense regular language, then $R \cap Q_\Sigma$ and $R - (R \cap Q_\Sigma)$ are disjunctive. Note that this result implies the following:

Proposition 7.1. *If $R \in \mathbf{REG}$ is dense, then R contains infinitely many non primitive words.*

This is because of the following reasons: let $R \in \mathbf{REG}$ be dense. By [6], $L := R - (R \cap Q_\Sigma)$ is disjunctive. In particular, Σ^*/\sim_L is an infinite set. If L is a finite set, then L is regular, so Σ^*/\sim_L is a finite monoid. Then Σ^*/\sim_L is a finite set, which is a contradiction. Thus, L is an infinite set, i.e., R contains infinitely many non primitive words, so we obtain Proposition 7.1.

Note that Proposition 7.1 is almost the same as Theorem 2.2 (and Theorem 1.1). The only difference is that Theorem 2.2 (and Theorem 1.1) tells us specific examples of non primitive words, i.e., R contains infinitely many non primitive words of the form z^{pn+1} , while Proposition 7.1 does not tell us such examples. As for Main Theorem 2.3, we have proved that if $L \in \Gamma(\mathbf{TL})$ is dense, then we have the condition (1), so there exists $z \in \Sigma^+$ and $p \geq 1$ such that $z^{pn+1} \in L$ ($\forall n \geq 0$). In addition, Main Theorem 2.3 is a non-trivial generalization of Theorem 2.2, as we have already proved in Section 5.

References

- [1] Berstel, Jean and Perrin, Dominique. *Theory of Codes*. Academic Press, 1985.
- [2] Berstel, Jean, Perrin, Dominique, and Reutenauer, Christophe. *Codes and Automata*, volume 129. Cambridge University Press, 2010.
- [3] Dömösi, Pál, Horváth, S, and Ito, M. Formal languages and primitive words. *Publ. Math. Debrecen*, 42(3-4):315–321, 1993.
- [4] Dömösi, Pál and Ito, Masami. *Context-free Languages and Primitive Words*. World Scientific, 2015.
- [5] Guo, YQ, Xu, GW, and Thierrin, Gabriel. Disjunctive decomposition of languages. *Theoretical Computer Science*, 46:47–51, 1986. DOI: 10.1016/0304-3975(86)90020-4.
- [6] Ito, Masami. Dense and disjunctive properties of languages. In *International Symposium on Fundamentals of Computation Theory*, pages 31–49. Springer, 1993. DOI: 10.1007/3-540-57163-9_3.
- [7] Koga, Toshihiro. On the density of regular languages. *Fundamenta Informaticae*, 168(1):45–49, 2019. DOI: 10.3233/FI-2019-1823.
- [8] Li, Zheng-Zhu, Shyr, Huei-Jan, and Tsai, YS. Classifications of dense languages. *Acta Informatica*, 43(3):173–194, 2006. DOI: 10.1007/s00236-006-0015-y.
- [9] Liu, YJ and Xu, ZB. Monoid algorithms and semigroup properties related to dense regular languages. In *Proceedings of the International Conference on Algebra and Its Applications, Bangkok*, pages 203–214. Citeseer, 2002.
- [10] Pin, Jean-Éric. *Mathematical Foundations of Automata Theory*, volume 7. 2010.
- [11] Reis, CM and Shyr, Huei-Jan. Some properties of disjunctive languages on a free monoid. *Information and Control*, 37(3):334–344, 1978. DOI: 10.1016/S0019-9958(78)90578-8.
- [12] Salomaa, Arto and Soittola, Matti. *Automata-Theoretic Aspects of Formal Power Series*. Springer Science & Business Media, 2012.
- [13] Shyr, HJ and Tseng, Din-Chang. Some properties of dense languages. *Soochow J. Math*, 10:127–131, 1984.
- [14] Shyr, Huei-Jan. Disjunctive languages on a free monoid. *Information and Control*, 34(2):123–129, 1977. DOI: 10.1016/S0019-9958(77)80008-9.
- [15] Sin'ya, Ryoma. An automata theoretic approach to the zero-one law for regular languages: Algorithmic and logical aspects. *Electronic Proceedings in Theoretical Computer Science*, 193:172–185, Sep 2015. DOI: 10.4204/eptcs.193.13.

- [16] Sin'ya, Ryoma. Asymptotic approximation by regular languages. In *SOFSEM 2021: Theory and Practice of Computer Science*, pages 74–88, Cham, 2021. Springer International Publishing. DOI: 10.1007/978-3-030-67731-2_6.
- [17] Sin'ya, Ryoma. Asymptotic approximation by regular languages, 2021. Online Worldwide Seminar on Logic and Semantics, <https://www.cs.bham.ac.uk/~vicaryjo/owls/slides/sinya.pdf>.

Received 14th June 2021

Models and Algorithms for Social Distancing in Order to Stop the Spread of COVID-19

Alexandru Popa^{ab}

Abstract

Currently there are many attempts around the world to use computers, smartphones, tablets and other electronic devices in order to stop the spread of COVID-19. Most of these attempts focus on collecting information about infected people, in order to help healthy people avoid contact with them. However, social distancing decisions are still taken by the governments empirically. That is, the authorities do not have an automated tool to recommend which decisions to make in order to maximize social distancing and to minimize the impact for the economy.

In this paper we address the aforementioned problem and we design an algorithm that provides social distancing methods (i.e., what schools, shops, factories, etc. to close) that are efficient (i.e., that help reduce the spread of the virus) and have low impact on the economy.

On short: a) we propose several models (i.e., combinatorial optimization problems); b) we show some theoretical results regarding the computational complexity of the formulated problems; c) we give an algorithm for the most complex of the previously formulated problems; d) we implement and test our algorithm.

Keywords: combinatorial optimization, COVID-19, algorithm, NP-hard problem

1 Introduction

The rapid spread of COVID-19 around the world is stunning. This novel coronavirus created an unprecedented lockdown in many countries which, in turn, caused an immense economic and social impact. Thus, many researchers investigate methods to stop this epidemic as soon as possible. For example, the list of papers on COVID-19 collected by the World Health Organization [28] contains around 10 000 publications, a huge number, given that the virus was first discovered in January 2020. The struggle involves researchers from various fields such as bioinformatics, epidemiology, sociology, mathematics and computer science.

^aDepartment of Computer Science, University of Bucharest, Romania

^bE-mail: alexandru.popa@fmi.unibuc.ro, ORCID: 0000-0003-3364-1210

One key factor to stop the spread of the virus is the *social distancing* (see, e.g., [10]). Many companies and organizations try to develop applications to aid the social distancing (see [25] for a long list of current such projects). However, many applications seem to focus on tracking people movement. To the best of our knowledge, we do not know any application that advises the authorities which decisions to make. As Thomas Pueyo writes in his article published on the 19th of March 2020 [20] (Chart 16), governments should have a chart with the effect and the cost of various social distancing measures. As it is currently observed in the world (and especially in Europe), many governments were afraid to take severe social distancing measures in order to avoid a high economic loss.

In this paper we try to address this issue as follows. We first build a model (i.e., a combinatorial optimization problem) that captures the current setting: the risk of the people to get COVID-19, the contact between various people and the cost of closing various facilities such as schools, parks, cities, factories, etc.. We show that the problem we introduce is NP-hard (as it is often the case with complex combinatorial optimization problems). Then, since we cannot solve the problem exactly in polynomial time, we provide a heuristic polynomial time algorithm for this problem. To understand the performance of our algorithm, we implement and test it in Section 5. We generate our test data using special probability distributions that simulate real world social networks as we present in Section 5.1. Our experiments are encouraging and show that even with a 1% budget (from the total cost of locking down the entire country), we can reduce the population risk by more than 5 times compared with the situation in which no measures are taken. Thus, we show that there is a possibility for a “beautiful” lockdown that is efficient in the fight with COVID-19 and safe for the economy.

1.1 Related work

In this paragraph we present briefly the related work in the field. Since the number of papers written on the topic is huge, it is impossible to mention all the results. Nevertheless, we enumerate a couple of papers that we consider relevant to the current work.

We first mention that some of the models presented in the paper are connected to random graphs. Random graphs is an active area of research which combines probability theory and graph theory. The subject began in 1960 with the seminal paper of Erdős and Rényi [6]. The book by Bollobás [5] is the standard source for the field. Other notable sources are [11, 7, 15, 2].

We now briefly mention the role of operations research in developing epidemic response strategies. One of the main applications of operations research was health-care, e.g. [21, 12, 30, 24, 4, 18]. However, most of the papers in the operations research field are concerned with either simulation frameworks [27, 23, 16] or resource allocation problems [22, 17, 13, 3].

A few models in the literature attempted to evaluate and devise response strategies among which we mention [26, 14, 31]. For more details, we refer the reader to the survey of Yu et al. [29], the related work section in the paper of Gillis et al. [9],

and the survey of Adiga et al. [1].

We mention that the paper is written in a bottom-up fashion. More precisely, in Section 2 we present a preliminary model that we designed in the early stages of our study. Even if we do not consider the model in Section 2 further in the paper, the motivation for introducing it is two-fold. Firstly, by presenting the model in Section 2 we show the reader the complete path we took to design the model in Section 3 (instead of simply presenting the final product). Secondly, researchers who aim to study and improve the models presented in this paper may find useful to understand the difficulty behind designing a comprehensive model.

The paper is structured as follows. At first, in Section 2 we present the first set of problems that aim to model the problem. We also show that these problems are NP-hard. Then, in Section 3 we present our actual framework. In Section 4 we design an algorithm for the problem presented in Section 3. Then, in Section 5 we describe our experiments. Finally, in Section 6 we discuss several directions for future work.

2 Preliminary ideas

In this section we introduce a preliminary model (i.e., a collection of related combinatorial optimization problems) that helped us to derive the model from Section 3.

2.1 A tentative framework

The input consists of an undirected *complete* graph $G = (V, \binom{V}{2})$ and a function $p : \binom{V}{2} \rightarrow [0, 1]$. Each node $v \in V$ in the graph corresponds to a person and $p(u, v)$ is the probability that two people get in contact with each other. Moreover, each vertex $v \in V$ has two associated values, *risk* : $V \rightarrow [0, 1]$ and *vulnerability* : $V \rightarrow [0, 1]$, representing how likely is a person to spread the disease (e.g., it can be 1 if a person is tested positive with COVID-19 or close to 1 if a person was recently in a “red area”), respectively how vulnerable is a certain person (e.g., there are studies showing that elderly people and people with chronic diseases are more likely to be affected).

Besides the input graph we are given k_1 sets of vertices $V^1 = \{V_1^1, V_2^1, \dots, V_{k_1}^1\}$ each one having associated a value $c_1 : \{1, 2, \dots, k_1\} \rightarrow \mathbb{R}_+$ and a value $r_1 : \{1, 2, \dots, k_1\} \rightarrow [0, 1]$. The cost $c_1(i)$ represents the cost of reducing the value of all $p(a, b)$, $\forall a, b \in V_i^1$ to $p(a, b) \cdot r_1(i)$. Informally, the cost $c_1(i)$ represents the cost of closing facility i (i.e., a school, a bar, restaurant, theater, etc.), which in turn reduces the probability of interaction of people belonging to the corresponding facility. In a simple variant, each $r_1(i)$ can be set to 0, representing that two people who belong to that facility will have probability 0 to interact once the facility is closed.

Then, we have k_2 sets of vertices $V^2 = \{V_1^2, V_2^2, \dots, V_{k_2}^2\}$ each one with a value $c_2 : \{1, 2, \dots, k_2\} \rightarrow \mathbb{R}_+$ and a value $r_2 : \{1, 2, \dots, k_2\} \rightarrow [0, 1]$. The cost $c_2(i)$ represents the cost of reducing the value of all $p(a, b)$ to $p(a, b) \cdot r_2(i)$, where $a \in V_i^2$

and $b \notin V_i^2$. Informally, the cost $c_2(i)$ represents the cost of isolating the people in the group V_i^2 (for example, quarantining persons, small groups or even closing entire cities).

2.2 Possible combinatorial optimization problems

Now we introduce a couple of objective functions and constraints that aim to model the current scenario. The overall goal is to reduce the spread of the virus while keeping the cost at a minimum. The first group of problems consider a simplified variant of the framework, ignoring the vulnerability and the risk of each person.

In the first problem the goal is to optimize the economic cost of closing various facilities and isolating various groups of people, while maximizing the number of components created.

Problem 1. *We are given a budget $B \in \mathbb{R}_+$ and a threshold $P \in [0, 1]$. The goal is to select a set $\hat{V}^1 \subseteq V^1$ and a set $\hat{V}^2 \subseteq V^2$ such that the following two conditions are met:*

1.

$$\sum_{i \in \hat{V}^1} c_1(i) + \sum_{i \in \hat{V}^2} c_2(i) \leq B$$

2. *After the sets of facilities \hat{V}^1 and \hat{V}^2 are selected and the corresponding edges have their probabilities decreased (as described in Subsection 2.1), we remove all the edges $(a, b) \in \binom{V}{2}$ such that $p(a, b) \leq P$. The goal is to maximize the number of connected components in the remaining graph.*

As we stated above, the model does not consider all the information. However, it is useful in cases where not much data is available to conduct preliminary tests. Moreover Problem 1 is interesting to study from the theoretical point of view since it is a novel combinatorial optimization problem.

Notice that even this oversimplified variant of the framework is NP-hard since it is a generalization of the classical Vertex Cover problem as we show in Subsection 2.3.

The second problem that we introduce is similar to the first problem. Here the goal is to minimize the budget, while requiring for at least a certain number of connected components to be created.

Problem 2. *We are given a number of desired connected components N and a threshold $P \in [0, 1]$. The goal is to select a set $\hat{V}^1 \subseteq V^1$ and a set $\hat{V}^2 \subseteq V^2$ such that the following holds. After the sets of vertices \hat{V}^1 and \hat{V}^2 are selected and the corresponding edges have their probabilities decreased, we remove all the edges $(a, b) \in \binom{V}{2}$ such that $p(a, b) \leq P$. The number of connected components in the remaining graph should be at least N . The goal is to minimize*

$$\sum_{i \in \hat{V}^1} c_1(i) + \sum_{i \in \hat{V}^2} c_2(i)$$

If we ask to maximize only the number of connected components we might obtain a solution that does not match the original motivation. For example, we can obtain a solution where we have many small components and a huge component, which is, of course, not desired in practice. Thus, we introduce the following two problems, in which we impose a restriction on the size of the connected components resulted after the closure of facilities.

Problem 3. *The input is the same as in Problem 1. The goal is to minimize the number of nodes of the largest connected components in the remaining graph.*

Problem 4. *The input is the same as in Problem 2, but N , instead of being the number of connected components desired, is the maximum allowed size of a connected component. Thus, the goal is to choose a set of facilities of minimum total budget (if such a set exists) such that, after closing these facilities, each resulting component has size less than or equal to N .*

In the end of this section, we formulate two more complex problems that aim to take into considerations all the restrictions, including the *risk* and the *vulnerability*.

Problem 5. *Besides the input graph and the data associated with the facilities, we are given a budget B , a threshold P and two real numbers W and R . We have the following constraints associated with the connected components resulted after closing the facilities:*

1. *For any connected component X we have $\sum_{v \in X} \text{vulnerability}(v) \leq W$. Informally, this constraint aims to avoid large groups formed by vulnerable people (such as elderly, or immunosuppressed).*
2. *For any connected component X we have $\sum_{u, v \in X} (\max\{\text{risk}(u), \text{risk}(v)\} - \min\{\text{risk}(u), \text{risk}(v)\}) \leq R$. Informally, this constraint aims to avoid a connected component that mixes “healthy” and “ill” people. Notice that if two people have high risk (i.e., that are very likely to have COVID-19) or if two people have very low risk, then $\max\{\text{risk}(u), \text{risk}(v)\} - \min\{\text{risk}(u), \text{risk}(v)\}$ is very close to 0.*

The goal is to select a set of facilities such that, after removing the edges with probability less than P , minimises the number of connected components that violate any of the two above mentioned constraints.

The final problem that we propose in this section, is very similar to Problem 5 but aims to enforce that all the components resulted obey the restrictions. Nevertheless, in this variant, we are not given a constraint on the budget. Otherwise, if we are given a constraint on the budget and on the connected components, it is NP-hard even to decide if a feasible solution exists (we obtain an instance of the Knapsack problem that is NP-hard [8]).

Problem 6. *The input is similar to Problem 5, except that we do not have a budget B . The goal is to select a set of facilities of minimum cost (if such a set exists) such that, after removing the edges with probability less than P , all the connected components do not violate any of the two constraints defined in Problem 5.*

2.3 Hardness results

In this section we show that the problems introduced in Subsection 2.2 are NP-hard. We show a complete proof only for Problem 1, since the NP-hardness proofs for the other problems are similar.

Theorem 1. *Problem 1 is NP-hard.*

Proof. We show a simple reduction from the Vertex Cover problem which is a classical NP-hard problem [8]. In the (decision version of the) Vertex Cover problem the input is an undirected graph $G = (V, E)$ and an integer k and the goal is to decide, if exists, a subset $V' \subseteq V$ such that $|V'| \leq k$ and for any edge $(a, b) \in E$, either $a \in V'$ or $b \in V'$ or both. Thus, given an instance of Vertex Cover, that is, a graph $G = (V, E)$ and an integer k , we construct an instance of Problem 1 as follows.

1. The input graph G' of Problem 1 has the same vertex set V .
2. The edge set is constructed as follows: for every edge $(a, b) \in E$, we set $p(a, b) = 1$, otherwise we set $p(a, b) = 0$.
3. We let $V^1 = \emptyset$.
4. We let $V^2 = \{\{v\} \mid \forall v \in V\}$, while the cost c_2 of selecting any set from V^2 is 1 and r_1 is 0 (that is, all the edges that are incident to a selected vertex are deleted).
5. The budget $B = k$.

Now, we show that the graph $G = (V, E)$ has a vertex cover of size at most k if and only if the maximum number of connected components in the corresponding instance of Problem 1, *after removing the edges (a, b) with $p(a, b) = 0$* , is n .

First, given a vertex cover V' , the solution of Problem 1 that creates n connected components selects the set $\hat{V}^2 = \{\{v'\} \mid v' \in V'\}$, that is, we select the sets from V^2 corresponding to the vertices in V' . Since V' is a vertex cover, any edge is incident to at least one vertex from V' , thus $p(a, b) = 0, \forall a, b \in V$ after selecting \hat{V}^2 .

Conversely, given a set \hat{V}^2 , such that $|\hat{V}^2| \leq k$, we construct the set $V' = \{v' \mid \{v'\} \in \hat{V}^2\}$. Since n connected components are created after selecting \hat{V}^2 , we know that $p(a, b) = 0, \forall a, b \in V$ (otherwise, we have a connected component with at least two vertices). Since $p(a, b) = 0, \forall a, b \in V$, we know that for any edge (a, b) that had $p(a, b) = 1$, either $\{a\} \in \hat{V}^2$ or $\{b\} \in \hat{V}^2$. Thus, $V' = \{v' \mid \{v'\} \in \hat{V}^2\}$ is a vertex cover of G , completing the proof. \square

Using a similar reduction, we can show that Problems 2, 3, 4, 5 and 6 are NP-hard. Thus, we state the following corollary.

Corollary 1. *Problems 2, 3, 4, 5 and 6 are NP-hard.*

3 The framework for modeling COVID-19

The framework presented in the previous section, although promising, has the following problem. The closure of a facility might not have the same effect for all the people that are connected through that facility. Consider the following simple example: two siblings (who live in the same house) study at the same school. Then, after closing the school, in reality the two siblings still have a large probability to get in contact with each other. Thus, we introduce the following framework which captures the aforementioned example and is also simpler than the framework presented in Section 2.

Problem 7. *The input consists of a bipartite graph $G = (U \cup V, E)$. The set U represents the people and the set V represents the facilities. For each edge we have associated a value $t : U \times V \rightarrow [0, 1]$ that represents the percentage of the time spent by a person in that facility in a day. For example, if $t(a, b) = 0.25$, then person a spends 6 hours (0.25×24 hours) in facility b . Each person has an associated probability $f : U \rightarrow [0, 1]$ of being infected. Each facility has an associated closure cost $c : V \rightarrow \mathbb{R}_+$. Closing a facility v is equivalent to removing the edges incident to v . Moreover, we are given a cost $c' : U \rightarrow \mathbb{R}_+$ of isolating people. Isolating a subset of people U' is equivalent to removing the edges incident to all the vertices in U' . Moreover, we are given a total budget B for closing the facilities.*

The risk of a facility is informally the weighted (using the probability of a person being infected f as the weight) sum of the time spent by the people in that facility. More precisely, $R : V \rightarrow \mathbb{R}_+$ is:

$$R(v) = \sum_{u \in U: (u,v) \in E} f(u) \cdot t(u, v)$$

The risk of a person $r : U \rightarrow \mathbb{R}_+$ (not to be confused with f) is defined as the weighted sum spent by a person in the facilities he visits (weighted using the risks of the facility). Formally:

$$r(u) = \sum_{v \in V: (u,v) \in E} R(v) \cdot t(u, v)$$

We define $r(U)$ the vector in $\mathbb{R}^{|U|}$, that has in each component the risk of a person.

The goal is to select a set of facilities of total cost at most B such that a given function $F : r(U) \rightarrow \mathbb{R}$ is minimized. In this paper we study the case when F is the ℓ_1 metric. In other words, we aim to optimize the total risk of the people.

We show that Problem 7 is NP-hard even in an extremely restricted version in which there is only one person associated with each facility. Nevertheless, notice that unlike Problem 6, Problem 7 admits a trivial feasible solution. Thus, in Section 4 we tackle the problem via a heuristic algorithm and show that it gives promising results.

Theorem 2. *Problem 7 is NP-hard in the case $F = \ell_1$.*

Proof. We prove NP-hardness of Problem 7 via a reduction from the Subset Sum problem, defined as follows. In the Subset Sum problem the input is a set S of integers and an integer B and the goal is to decide if there exists a subset of integers from S whose sum is precisely B . The Subset Sum problem is a famous NP-hard problem [8]. Given an instance of the Subset Sum problem, we create an instance of Problem 7 as follows.

For each $x \in S$, we create a facility v of cost $c(v) = x$. Thus, V includes these v vertices. The bipartite graph $G = (U \cup V, E)$ with vertex classes U and V is a balanced one, that is, $|U| = |V|$, and the edge set of G is a perfect matching $M = E$. In other words, every vertex of G has degree one.

For every $u \in U$ we let $f(u) = \frac{c(v)}{\max_{w \in V} c(w)}$ and $t(u, v) = 1$, where v is the only neighbor of u , that is $(u, v) \in M$. Thus, for each pair person/facility $(u, v) \in M$ we have $R(v) = r(u) = f(u)$. Note that if a set of facilities (some subset of V) is closed with cost X , then the total risk is

$$\frac{\sum_{w \in V} c(w) - X}{\max_{w \in V} c(w)}.$$

The above shows that the total risk is

$$\frac{\sum_{w \in V} c(w) - B}{\max_{w \in V} c(w)}$$

if and only if there exist a subset of numbers from S that have sum precisely B .

Thus, Problem 7 is NP-hard in the case $F = \ell_1$. \square

4 The algorithm

In this section we provide a heuristic (approximation) algorithm for Problem 7. We test our algorithm in Section 5 and show that it gives promising results.

Our algorithm (presented in Algorithm 1) sorts the list of people and the facilities according to their efficiency (the cost of isolating/closing a person/facility divided by the amount of risk the people/facilities have). Then, the algorithm aims to find the optimum division of the available budget between isolating people and closing facilities. According to our experiments (see Section 5) there is not an obvious correlation between the optimal value of the division of the budget (i.e., variable *Split* in Algorithm 1) and the minimum total risk. Thus, we need to iterate over all values of *Split* in order to find a good solution. Of course, since there are infinitely many numbers between 0 and 1, we cannot iterate over all possible values. Choosing a larger increment improves the running time but reduces the accuracy of the solution.

1. Define the efficiency of a facility v as

$$e(v) = \frac{c(v)}{R(v)}$$

2. Define the efficiency of isolating a person u as

$$e'(u) = \frac{c'(u)}{f(u)}$$

3. Sort the sequence of values e and e' in increasing order.

4. $MinRisk = \infty$

5. For every value of $Split$ between 1 and 100 (in increments of 1) do:

- a) Isolate people in the order given by e' until a budget of $B \cdot \frac{1}{Split}$ is reached.
- b) Close the facilities in the order given by e until the budget B is reached.
- c) Let R_{Split} be the total risk of the population according to this solution. If $R_{Split} < MinRisk$ then we update the value of $MinRisk$ and store the current solution.

6. **Output:** $MinRisk$ and the corresponding set of people and facilities that have to be isolated/closed.

Algorithm 1: A heuristic algorithm for Problem 7.

5 Experiments

5.1 Data generation

In this subsection we describe how we generated our data.

First our data generator allows two parameters as input that determine the number of facilities and the maximum size of a facility. The size of the facilities (i.e., how many people visit that facility in a day) is drawn according to a power law distribution with exponent α (in our experiments α varies between 0.8 and 1.3). We also select an average number of daily activities for a person (i.e., how many facilities a person visits during one day). In our experiments the average number of activities is set between 3 and 8). Then, we set the number of people in a country to be the sum of all the facilities divided by the average number of facilities a person visits during one day.

In Figure 1 we show an example of the distribution of the size of the facilities for 1000 facilities each having a size between 10 and 10000.

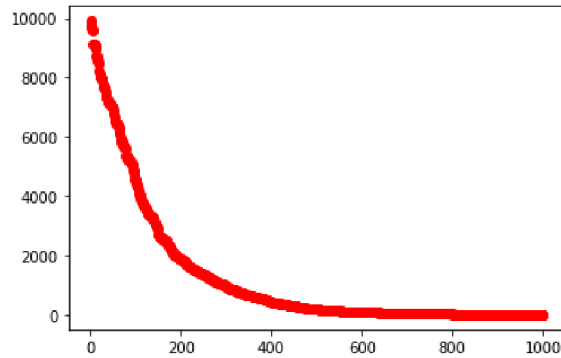


Figure 1: The size of 1000 facilities (i.e. daily number of people that visit that facility). Each facility has at least 10 and at most 10000 daily visitors. The number of visitors is drawn from a power law distribution with $\alpha = 1.1$.

For each facility v we select $size(v)$ people that will visit that facility uniformly at random from the population, where $size(v)$ is the size of facility v that was generated previously using the power law distribution. The number of activities performed daily by each person form a Poisson distribution (see Figure 2 for an example).

We now show how we generate the weights on the edges. For each person, we choose the time spent in each facility using an exponential distribution.

The probability that a person i carries the virus, i.e., $f(i)$, is also drawn from a power law distribution with exponent α_2 . One important thing to notice is that

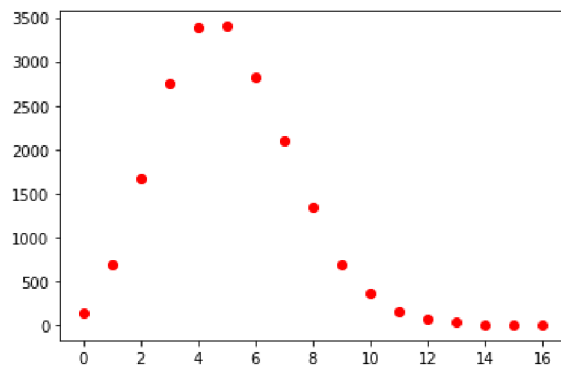


Figure 2: The distribution of number of daily activities for a population of 19573 people that have on average 5 daily activities.

α_2 influences significantly the risk of the whole population to get infected. More precisely, if α_2 is large (that is, there are few people with high risk of carrying the virus), the risk of infection for the other people is relatively low. In Figure 3 we show the risk associated to the people (calculated as shown in Problem 7) for the values of $\alpha_2 = 4$ and $\alpha_2 = 2$. This observation motivates us in the design of algorithm by isolating first the persons with very high risk.

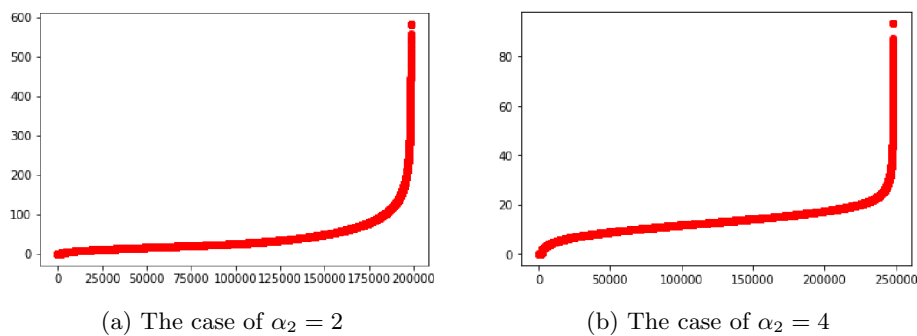


Figure 3: Comparison of the risk of the population to get infected for $\alpha_2 = 2$ and $\alpha_2 = 4$

Finally, we have to set the cost of isolating people and the cost of closing facilities. We choose the cost of isolating a person as a fraction of total budget available (this fraction can also be set as an input parameter in our generator). The cost of closing a facility of size s is s^x , where x is a random variable drawn according to a Gaussian distribution with mean μ and variance σ (in our tests we vary the μ between 1.1 and 1.2 and σ between 0.3 and 0.5). Finally, the budget is also an input parameter in the generator and we design it as a fraction of the total cost of closing the facilities, generally, between 1% and 30%.

5.2 Tests

We carried out tests for a population of around 30 000 people. This population is achieved by varying the parameters in our model as: the number of facilities (between 100 and 1 000), the average number of daily activities (between 3 and 8) and the size of each facility (between 4 and 10 000). For each set of parameters we carried out 5 tests and we chose the average risk produced by our algorithm over these 5 tests.

The dataset size is the maximum that our hardware can handle. Nevertheless, we argue that our experiments scale to a larger population. In Figure 4 we show how the risk changes if we change the number of facilities and the size of each facility: the risk has a decreasing trend as the size of our population increases, thus we believe that our algorithm is even better for larger scale instances.

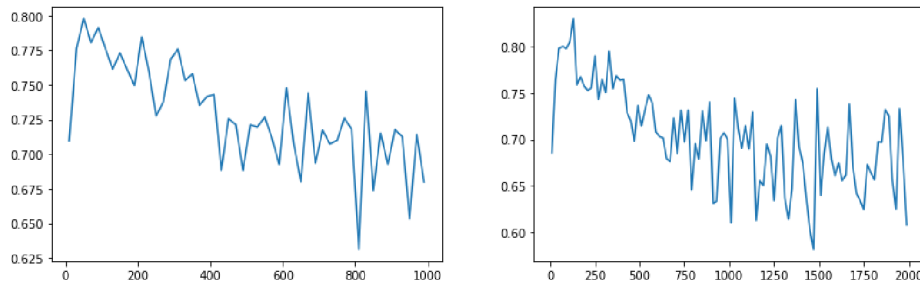
Next, we show how the split of the budget between isolating people and closing facilities influences the total risk. In our tests we have 500 facilities between 4 and

1 000 people, each person performs on average 4 activities per day and we have $\alpha = 1.1$, $\alpha_2 = 2$. The cost of the exponent of the random variable that determines the cost of closing the facilities is drawn from a normal distribution with $\mu = 1.1$ and $\sigma = 0.4$ (Figure 5a and Figure 6a), $\sigma = 0.5$ (Figure 5b and Figure 6b). The budget is 10% of the cost of closing all facilities in Figure 5 and 1% in Figure 6. This budget suffices to isolate 10% of the population, respectively 1%.

Notice that with a budget of only 1% from the cost of closing all facilities, we are able to lower the risk to less than 20% of the original risk (Figure 6b).

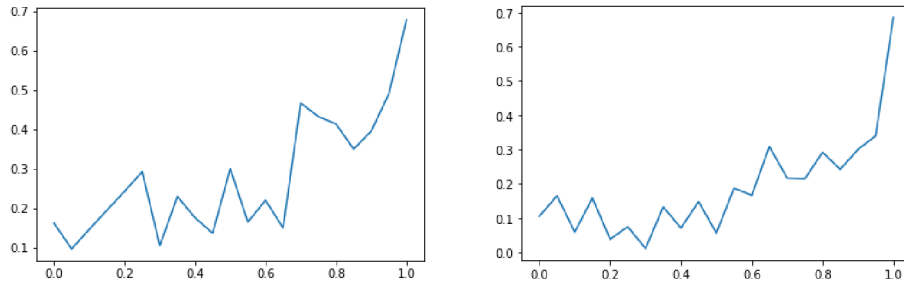
Finally, we tested how does the risk decrease if we take actions quickly. More precisely, we vary α_2 which is the power law exponent that determines the percentage of people that are likely to be already infected. However, we did not notice any major influence of this factor in the total risk if the infection proportion is drawn according to a power law distribution.

Our algorithm was implemented in Python and the tests were carried out on a 2013 MacBook Pro with 2.4 GHz Quad-Core Intel Core *i7*, and 8GB RAM. The code used for testing and generating data is available on GitHub [19].



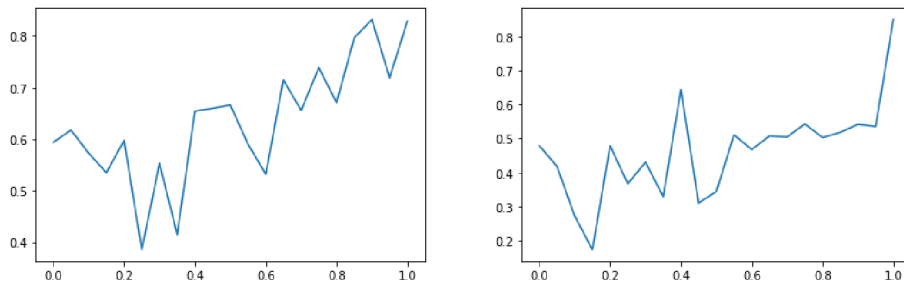
(a) The horizontal axis represents the number of facilities, while the vertical axis represents the risk improvement. The size of each facility is between 4 and 1 000. (b) The horizontal axis represents the size of each facility, while the vertical axis represents the risk improvement. The number of facilities is 500 and the average number of daily activities performed by a person is 4.

Figure 4: The improvement in the population risk (i.e., the risk of the population after our algorithm, divided by the risk before the run of our algorithm) compared with the size of the instance. The average number of daily activities performed by a person is 4. The budget allocated is 5% of the amount necessary to close all facilities. With this budget we are able to quarantine 5% of the population. Then, we have $\alpha = 1.1$ and $\alpha_2 = 2$. Observe that the improvement is bigger as the population increases.



(a) A 10% of the total cost of closing the facilities and $\sigma = 0.4$ (b) A 10% of the total cost of closing the facilities and $\sigma = 0.5$

Figure 5: On the x axis is the percentage of the total budget allocated to isolating people. On the y axis there is the ratio of the risks before/after running the algorithm.



(a) A 1% of the total cost of closing the facilities and $\sigma = 0.4$ (b) A 1% of the total cost of closing the facilities and $\sigma = 0.5$

Figure 6: On the x axis is the percentage of the total budget allocated to isolating people. On the y axis there is the ratio of the risks before/after running the algorithm.

6 Conclusions and future work

In this paper we presented a model and an algorithm that aims to help authorities to take more efficient decisions in the fight with COVID-19. Naturally, the most stringent open problem is to test and validate the model and the algorithm on real data. People have a huge mobility nowadays and it is impossible to create a model which is fully accurate. Nevertheless, based on our tests we believe that our model is capable of capturing the most important features of the current situation.

Also, a natural open problem is to tune the input parameters: the probabilities p in the input graph, the cost of closing facilities and isolating people c and c' and the contagion risk associated with each person.

Since the appearance of vaccines the strategy for tackling COVID has changed significantly. Nevertheless, in some countries, restrictions are still in place. Thus, we are hopeful that our model will give the authorities some insight in taking the best decisions. Moreover, as we can see from our experiments, even with a very small budget (sometimes as low as 1% of the total cost necessary to lock down the entire economy), the risk of infection can be decreased significantly. Thus, we strongly believe that, with wise decisions, it is possible to stop the spread of COVID-19 and future pandemics without an economic collapse.

Acknowledgements

I would like to thank Ramona Georgescu, Péter Biró, Radu Mincu and Lucian-Ionut Gavrilă for extremely useful discussions.

References

- [1] Adiga, Aniruddha, Dubhashi, Devdatt, Lewis, Bryan, Marathe, Madhav, Venkatramanan, Srinivasan, and Vullikanti, Anil. Mathematical models for COVID-19 pandemic: A comparative analysis. *Journal of the Indian Institute of Science*, pages 1–15, 2020. DOI: 10.1007/s41745-020-00200-6.
- [2] Alon, Noga and Spencer, Joel H. *The Probabilistic Method*. Wiley, New York, second edition, 2004. DOI: 10.1002/0471722154.
- [3] Bertsimas, Dimitris, Ivanhoe, Joshua Kiefer, Jacquillat, Alexandre, Li, Michael Lingzhi, Previero, Alessandro, Lami, Omar Skali, and Bouardi, Hamza Tazi. Optimizing vaccine allocation to combat the COVID-19 pandemic. *medRxiv*, 2020. DOI: 10.1101/2020.11.17.20233213.
- [4] Biró, Péter, Manlove, David F., and Rizzi, Romeo. Maximum weight cycle packing in directed graphs, with application to kidney exchange programs. *Discret. Math. Algorithms Appl.*, 1(4):499–518, 2009. DOI: 10.1142/S1793830909000373.
- [5] Bollobás, Béla. *Modern Graph Theory*, volume 184 of *Graduate Texts in Mathematics*. Springer, 2002. DOI: 10.1007/978-1-4612-0619-4.
- [6] Erdos, Paul, Rényi, Alfréd, et al. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, 5(1):17–60, 1960.
- [7] Frieze, Alan and Karoński, Michał. *Introduction to Random Graphs*. Cambridge University Press, 2015. DOI: 10.1017/CB09781316339831.

- [8] Garey, M. R. and Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, first edition, 1979.
- [9] Gillis, Melissa, Urban, Ryley, Saif, Ahmed, Kamal, Noreen, and Murphy, Matthew. A simulation–optimization framework for optimizing response strategies to epidemics. *Operations Research Perspectives*, 8(C), 2021. DOI: 10.1016/j.orp.2021.100210.
- [10] Glass, Robert J., Glass, Laura M., Beyeler, Walter E., and Min, H. Jason. Targeted social distancing designs for pandemic influenza. *Emerging Infectious Diseases*, 12(11):1671, 2006. DOI: 10.3201/eid1211.060255.
- [11] Janson, Svante, Luczak, Tomasz, and Rucinski, Andrzej. *Random graphs*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley, 2000. DOI: 10.1002/9781118032718.
- [12] Kahraman, Cengiz and Topcu, Y Ilker. *Operations Research Applications in Health Care Management*. Springer, 2018. DOI: 10.1007/978-3-319-65455-3.
- [13] Kaplan, Edward H. Containing 2019-nCoV (Wuhan) coronavirus. *Health Care Management Science*, 23(3):311–314, 2020. DOI: 10.1007/s10729-020-09504-6.
- [14] Keimer, Alexander and Pflug, Lukas. Modeling infectious diseases using integro-differential equations: Optimal control strategies for policy decisions and applications in COVID-19. Technical Report, 2020.
- [15] Kolchin, V. F. *Random Graphs*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1998. DOI: 10.1017/CB09780511721342.
- [16] Malone, John D, Brigantic, Robert, Muller, George A, Gadgil, Ashok, Delp, Woody, McMahon, Benjamin H, Lee, Russell, Kulesz, Jim, and Mihelic, F Matthew. US airport entry screening in response to pandemic influenza: Modeling and analysis. *Travel Medicine and Infectious Disease*, 7(4):181–191, 2009. DOI: 10.1016/j.tmaid.2009.02.006.
- [17] Mehrotra, Sanjay, Rahimian, Hamed, Barah, Masoud, Luo, Fengqiao, and Schantz, Karolina. A model of supply-chain decisions for resource sharing with an application to ventilator allocation to combat COVID-19. *Naval Research Logistics (NRL)*, 67(5):303–320, 2020. DOI: 10.1002/nav.21905.
- [18] Mincu, Radu Stefan, Biró, Péter, Gyetvai, Márton, Popa, Alexandru, and Verma, Utkarsh. IP solutions for international kidney exchange programmes. *Central Eur. J. Oper. Res.*, 29(2):403–423, 2021. DOI: 10.1007/s10100-020-00706-5.

- [19] Popa, Alexandru. A decision support system for optimizing the cost of social distancing in order to stop the spread of COVID-19. <https://github.com/alexpopa9/ResearchCode/blob/master/COVID-19.py>, 2020.
- [20] Pueyo, Thomas. Coronavirus: The hammer and the dance. <https://medium.com/@tomaspueyo/coronavirus-the-hammer-and-the-dance-be9337092b56>, 2020. [Online; accessed 06-04-2020].
- [21] Rais, Abdur and Viana, Ana. Operations research in healthcare: A survey. *International Transactions in Operational Research*, 18(1):1–31, 2011. DOI: 10.1111/j.1475-3995.2010.00767.x.
- [22] Risanger, Simon, Singh, Bismark, Morton, David, and Meyers, Lauren Ancel. Selecting pharmacies for COVID-19 testing to ensure access. *Health Care Management Science*, pages 1–9, 2021. DOI: 10.1007/s10729-020-09538-w.
- [23] Roche, Benjamin, Drake, John M, and Rohani, Pejman. An agent-based model to study the epidemiological and evolutionary dynamics of influenza viruses. *BMC Bioinformatics*, 12(1):1–10, 2011. DOI: 10.1186/1471-2105-12-87.
- [24] Roth, Alvin E., Sönmez, Tayfun, and Ünver, M. Utku. Kidney Exchange. *The Quarterly Journal of Economics*, 119(2):457–488, 05 2004. DOI: 10.1162/0033553041382157.
- [25] The Governance Lab. Data collaborative in response to COVID-19. https://docs.google.com/document/d/1JWeD1AaIGKMPry_EN8GjIqwX4J4KLQIAqP09exZ-ENI/edit#, 2020. [Online; accessed 06-04-2020].
- [26] Toda, Alexis Akira. Susceptible-Infected-Recovered (SIR) dynamics of COVID-19 and economic impact. *arXiv preprint arXiv:2003.11221*, 2020.
- [27] Wang, Weixing, Li, Yanli, and Zhang, Jinjin. System dynamics modeling of SARS transmission — a case study of Hebei Province. In *2009 International Conference on Management and Service Science*, pages 1–4, 2009. DOI: 10.1109/ICMSS.2009.5303731.
- [28] World Health Organization. Global research on coronavirus disease (COVID-19). <https://www.who.int/emergencies/diseases/novel-coronavirus-2019/global-research-on-novel-coronavirus-2019-ncov>, 2020. [Online; accessed 28-03-2020].
- [29] Yu, Shuo, Qing, Qing, Zhang, Chen, Shehzad, Ahsan, Oatley, Giles, and Xia, Feng. Data-driven decision-making in COVID-19 response: A survey. *IEEE Transactions on Computational Social Systems*, 8(4):1016–1029, 2021. DOI: 10.1109/TCSS.2021.3075955.
- [30] Zaric, Gregory S. *Operations Research and Health Care Policy*. Springer, 2013. DOI: 10.1007/978-1-4614-6507-2.

- [31] Zhigljavsky, Anatoly, Fesenko, Ivan, Wynn, Henry, Whitaker, Roger, Kremnizer, Kobi, Noonan, Jack, and Gillard, Jonathan. A prototype for decision support tool to help decision-makers with the strategy of handling the COVID-19 UK epidemic. *medRxiv*, 2020. DOI: 10.1101/2020.04.24.20077818.

Received 20th January 2021

CONTENTS

Uli Fahrenberg, Christian Johansen, Georg Struth, and Krzysztof Ziemiański:
Domain Semirings United 575

Gergely Szlobodnyik and Gábor Szederkényi: Computing Different Realizations of Linear Dynamical Systems with Embedding Eigenvalue Assignment 585

Tamás Jónás, Christophe Chesneau, József Dombi, and Hassan S. Bakouch:
The Inverse Epsilon Distribution as an Alternative to Inverse Exponential Distribution with a Survival Times Data Example 613

Sean Cleary and Roland Maio: An Efficient Sampling Algorithm for Difficult Tree Pairs 629

Munqath Alattar and Attila Sali: Strongly Possible Functional Dependencies for SQL 647

Andreas Rauh and Ekaterina Auer: Verified Integration of Differential Equations with Discrete Delay 677

Tamás Jónás: The Generalized Epsilon Function: An Alternative to the Exponential Function 703

Toshihiro Koga: Dense Languages and Non Primitive Words 717

Alexandru Popa: Models and Algorithms for Social Distancing in Order to Stop the Spread of COVID-19 733

ISSN 0324—721 X (Print)
ISSN 2676—993 X (Online)

Editor-in-Chief: Tibor Csendes