Time Granularities and Ultimately Periodic Automata

Davide Bresolin

Angelo Montanari

Gabriele Puppis

{bresolin,montana,puppis}@dimi.uniud.it

Dipartimento di Matematica e Informatica

Università degli Studi di Udine

Outline

- Motivation
- The notion of Time Granularity
- Approaches to Time Granularity
- The Automaton-based Approach:
  - Basic ingredients
  - Ultimately Periodic Automata (UPA)
- Paradigmatic problems and their solutions
- A Real-World Application
- Beyond UPA
- Further Work

Time Granularities - 1

Motivations:

- **Relational databases**:
  to express temporal information at different time granularities, to relate different granules and to convert associated data (queries)

- **Artificial intelligence**:
  to reason about temporal relationships, e.g, to check consistency and validity of temporal constraints at different time granularities (temporal CSPs)

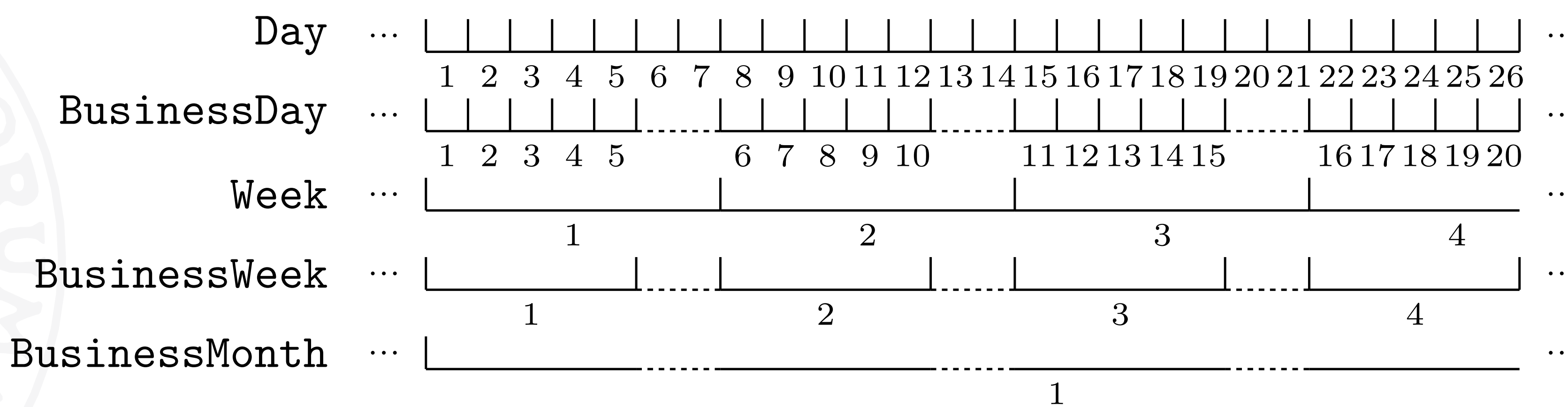
- **Specification and verification of reactive systems**:
  to specify and to check temporal properties of (real-time) reactive systems

Time Granularities - 2

Definition. $G : \mathbb{Z} \rightarrow 2^T$ is a *granularity* iff

- $(T, <)$ is a linearly ordered set of temporal instants,
- $t_x < t_y$ whenever $x < y$, $t_x \in G(x)$, and $t_y \in G(y)$.

A *granule* of G is a non-empty set $G(x)$ and $x \in \mathbb{Z}$ is said to be its *label*.



Approaches to Time Granularities - 1

Possible approaches to model time granularity:

- **algebraic**: it uses expressions built up from a set of symbolic operators

  (e.g., $\mathtt{Week} = Group_7(\mathtt{Day})$,
  cf. Bettini, Wang and Jajodia '00)

- **logical**: it identifies granularities with models of logical formulas

  (e.g., PLTL-formulas,
  cf. Combi, Franceschet and Peron '04)

Approaches to Time Granularities - 2

- **string-based**: it specifies time granularities through ultimately periodic strings over $\{\blacksquare, \square, \wr\}$

  (e.g., $(\blacksquare\blacksquare\blacksquare\blacksquare\blacksquare\square\square\wr)^\omega$ represents business weeks, cf. Wijsen '00)

- **automaton-based**: it exploits finite state automata (Büchi automata) to represent granularities that, ultimately, periodically group temporal instants

  (e.g., Single String Automata, cf. Dal Lago and Montanari '01)

The Automaton-based Approach - 1

We followed the automaton-based approach, trying to achieve

1. **expressiveness**, namely, to capture a large set of granularities
2. **compactness**, namely, to obtain size-optimal representations
3. **effectiveness**, namely to ease algorithmic manipulation, in particular w.r.t. the following fundamental problems:
   - **equivalence**, which consists in deciding whether two given automata represent the same granularity
   - **granularity comparison**, which consist in relating different temporal structures
   - **optimization**, which consists in manipulating representations in order to optimize the running time of crucial algorithms.

The Automaton-based Approach - 2

Basic ingredients:

- a discrete temporal domain T
- restriction to *left bounded periodical* granularities
- a fixed alphabet $\{\blacksquare, \square, \blacktriangleleft\}$, where

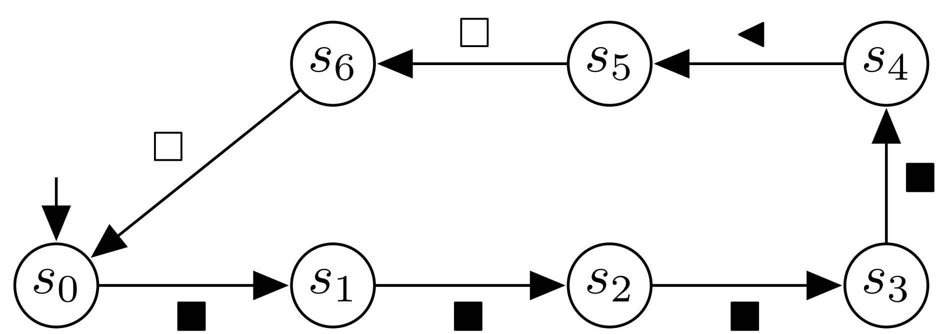
$\blacksquare$	represents elements covered by some granule,
$\square$	represents gaps within and between granules,
$\blacktriangleleft$	represents the last element of a granule.

Single String Automata

Proposition. *Ultimately periodic words* over $\{\blacksquare, \square, \blacktriangleleft\}$ capture all the left bounded periodical granularities.

Ultimately periodic words can be finitely represented by using Büchi automata recognizing *single words*.

$\Rightarrow$ notion of **Single String Automaton (SSA)**.



The SSA for the business-week granularity.

From Single Granularities to Sets of Granularities

We generalize the automaton-based approach to capture *sets* of granularities, instead of single time granularities, by means of larger subclasses of Büchi automata.

Remark. Büchi automata recognize ω*-regular languages*.

$\Rightarrow$ we started by considering sets of granularities which are represented by

ω**-regular languages of ultimately periodic words**.

Dealing with Sets of Granularities - 1

Proposition. An ω-regular language L consists of only ultimately periodic words iff it is a finite union of sets of the form

$$U \cdot \{v\}^{\omega}$$

with $U \subseteq \Sigma^*$ being a regular language and v a finite non-empty word.

$\Rightarrow$ We can represent sets of granularities featuring

- *a possibly infinite number of different prefixes*
- *a finite number of non-equivalent repeating patterns*

  (equivalent patterns are those which can be obtained by rotating and/or repeating a given finite word
  e.g. □■◀ and ■◀□■◀□)

Dealing with Sets of Granularities - 2

$\Rightarrow$ the notion of **Ultimately Periodic Automata** (UPA) comes into play.

UPA are Büchi automata where the strongly connected component of any final state is either a *single transient state* or a *simple loop* with no exiting transitions.

(each loop acts like an SSA recognizing a single periodic word)

$\Rightarrow$ UPA capture all and only the ω-regular languages of ultimately periodic words.

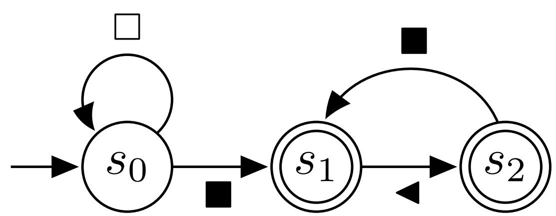
Remark. Such languages are closed under *union*, *intersection*, *concatenation* with regular languages, but not under *complementation*.

Dealing with Sets of Granularities - 3

Examples.

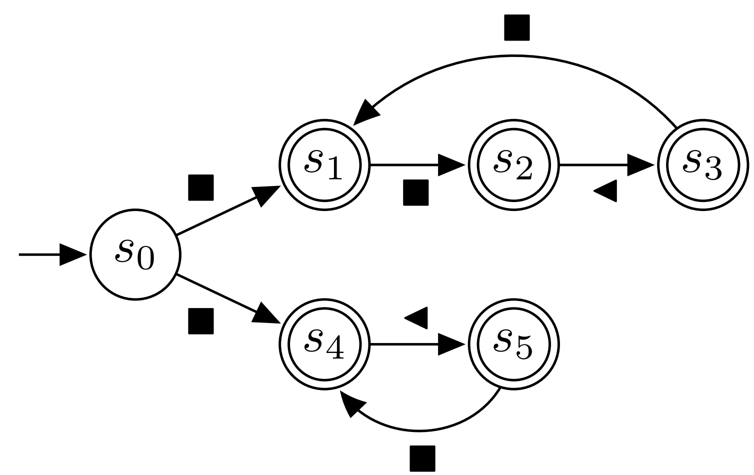
The set of granularities that groups days two-by-two:

$$\{\square\}^* \cdot \{\blacksquare \blacktriangleleft\}^\omega$$



The set of granularities that groups day either two-by-two or three-by-three:

$$\{\blacksquare\blacksquare \blacktriangleleft\}^\omega \cup \{\blacksquare \blacktriangleleft\}^\omega$$



Paradigmatic problems

Emptiness.
Decide whether the language of a given UPA is empty.

Membership.
Given an UPA $\mathcal{A}$ and a word w, decide whether $w \in \mathcal{L}(\mathcal{A})$.

Equivalence.
Decide whether two UPA recognize the same language.

Minimization.
Compute the smallest UPA recognizing a given language.

Granularity comparison.
For any pair of sets of granularities $\mathcal{G}, \mathcal{H}$, decide whether there exist $G \in \mathcal{G}$ and $H \in \mathcal{H}$ such that $G \sim H$, with $\sim$ being one of the usual relation between granularities (e.g., *finer than*, *groups into*, $\ldots$).

Emptiness, membership, and equivalence problems

Emptiness.

Solved in linear time by testing the existence of a reachable loop involving some final state.

Membership.

Given an UPA $\mathcal{B}$ recognizing $\{w\}$, test the emptiness of the language recognized by the product automaton $\mathcal{A} \times \mathcal{B}$ over the alphabet $\{\binom{\square}{\square}, \binom{\blacksquare}{\blacksquare}, \binom{\blacktriangleleft}{\blacktriangleleft}\}$.

Equivalence (Trivial Solution.)

Consider $\mathcal{A}$ and $\mathcal{B}$ as Büchi automata: compute their complements $\overline{\mathcal{A}}$ and $\overline{\mathcal{B}}$, and test the emptiness of both $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\overline{\mathcal{B}})$ and $\mathcal{L}(\mathcal{B}) \cap \mathcal{L}(\overline{\mathcal{A}})$.

The equivalence problem

Equivalence (Improved Solution.)

Compute a canonical form for $\mathcal{A}$ and $\mathcal{B}$, that is *unique* up to isomorphisms:

1. minimize the patterns of the recognized words and the final loops (using Paige-Tarjan-Bonic algorithm);
2. minimize the prefixes of the recognized words;
3. compute the minimum *deterministic* automaton for the prefixes of the recognized words;
4. build the canonical form by adding the final loops to the minimum automaton for the prefixes.

Minimization and Comparison problems

Minimization.

Replace step 3 in the canonization algorithm with the computation of a minimal *non-deterministic* automaton for the prefixes.

The problem is PSPACE-complete and it may yields to different solutions.

Comparison of granularities.

Can be reduced to the emptiness problem as follows:

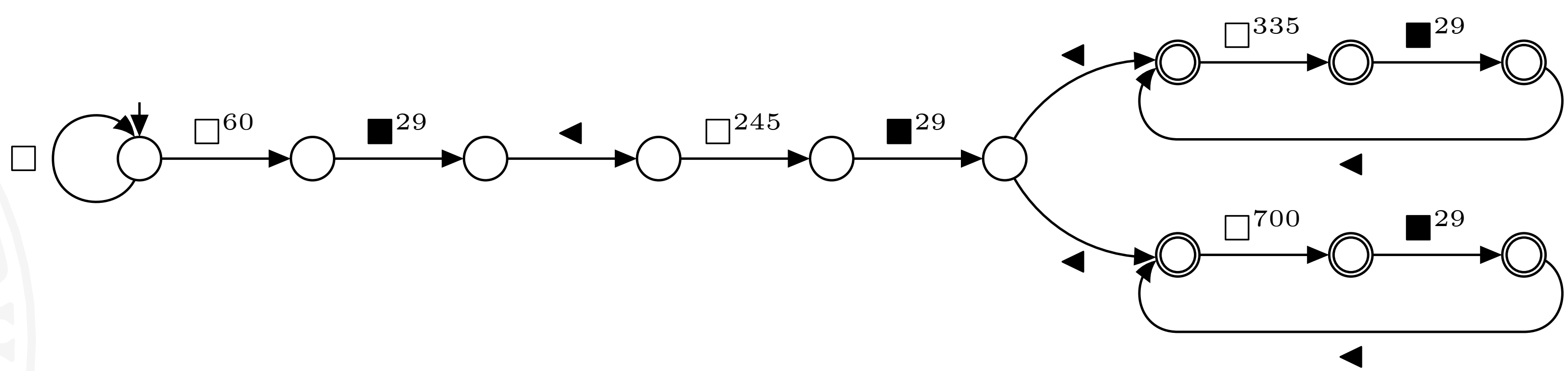
1. express the granularity relation in the string-based formalism;
2. define a product automaton that accepts all pairs of granularities that satisfy the relation;
3. test the emptiness of such an automaton.

A Real-World Application - 1

Posttransplantation guidelines: The patient must undertake a GFR estimation with one of the following schedule:

- 3 months, 12 months and every year thereafter;
- 3 months, 12 months and every 2 years thereafter.

$\Rightarrow$ UPA $\mathcal{A}$ representing the protocol:



A Real-World Application - 2

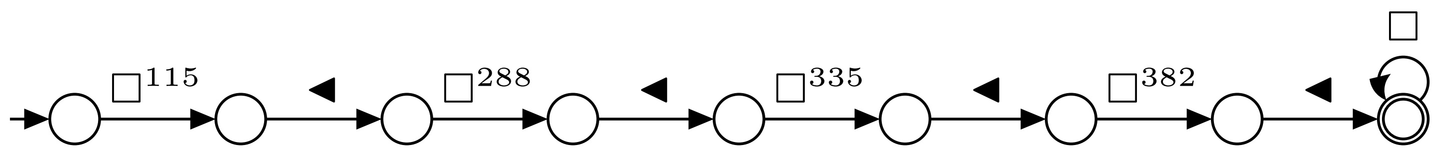
Consider the following instance of the temporal relation `VISITS(PatientId, Date, Treatment)`:

`PatientId`	`Date (MM/DD/YYYY)`	`Treatment`
1001	02/10/2003	transplant
1001	04/26/2003	GFR
1002	06/07/2003	GFR
1001	06/08/2003	biopsy
1001	02/10/2004	GFR
1001	01/11/2005	GFR
1001	01/29/2006	GFR

Problem: GFR measurement of patient 1001 respects the guidelines?

Solution to the problem - 1

Solution: Test whether the granularity of GFR measurement of patient 1001, represented by the UPA $\mathcal{B}$:



is an *aligned refinement* of some granularity recognized by $\mathcal{A}$.

Definition. A granularity G is an *aligned refinement* of the granularity H if, for every positive integer n, the n-th granule of G is included in the n-th granule of H.

Solution to the problem - 2

1. Given two words g and h, representing G and H, H is an aligned refinement of G iff, for every $n \in \mathbf{N}^+$:
   - $h[n] \in \{\blacksquare, \blacktriangleleft\} \;\Rightarrow\; g[n] \in \{\blacksquare, \blacktriangleleft\}$;
   - $h[1, n-1]$ and $g[1, n-1]$ encompass the same number of occurrences of $\blacktriangleleft$.

2. Given the UPA $\mathcal{A}$ for the protocol, and the UPA $\mathcal{B}$ for the visits, we can compute the product automaton for the aligned refinement relation.

Solution to the problem - 3

3. The product automaton recognizes the language:

$$
\begin{aligned}
\Big\{ & \tbinom{\square}{\square}^{100}\tbinom{\blacksquare}{\square}^{15}\tbinom{\blacksquare}{\blacktriangleleft}\tbinom{\blacksquare}{\square}^{13}\tbinom{\blacktriangleleft}{\square}\tbinom{\square}{\square}^{245}\tbinom{\blacksquare}{\square}^{29}\tbinom{\blacktriangleleft}{\blacktriangleleft}\tbinom{\square}{\square}^{335}\cdot \\
& \cdot\tbinom{\blacksquare}{\blacktriangleleft}\tbinom{\blacksquare}{\square}^{28}\tbinom{\blacktriangleleft}{\square}\tbinom{\square}{\square}^{335}\tbinom{\blacksquare}{\square}^{18}\tbinom{\blacksquare}{\blacktriangleleft}\tbinom{\blacksquare}{\square}^{10}\tbinom{\blacktriangleleft}{\square}\cdot \\
& \cdot\left(\tbinom{\square}{\square}^{335}\tbinom{\blacksquare}{\square}^{29}\tbinom{\blacktriangleleft}{\square}\right)^{\omega} \Big\}
\end{aligned}
$$

$\Rightarrow$ GFR measurements for patient 1001 respects the protocol guidelines.

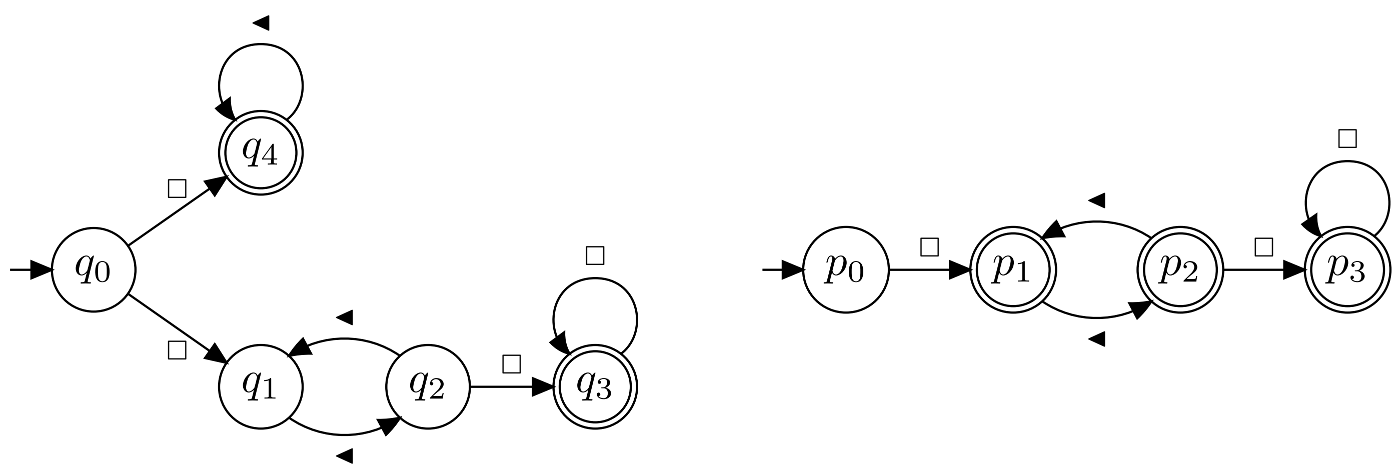
Redundancies in UPA

Problem: How to build compact representations of set of granularities?

$\Rightarrow$ we have an algorithm to minimize UPA.

But.. UPA may present redundancies in their structure:

- final and non-final loops that encodes the same patterns.



Relaxed UPA (RUPA)

Solution:

- Allow transitions to exit from final loops;
- whenever an automaton leaves a final loop, it cannot reach it again.

$\Rightarrow$ the notion of **Relaxed UPA (RUPA)** comes into play:

- These are Büchi automata where the SCC of any final states is either a *single transient state* or a *simple loop*.

Theorem. RUPA recognize all and only the UPA-recognizable languages.

Remark. UPA can be transformed into more compact RUPA by collapsing redundant final loops.

Beyond (R)UPA - 1

Open Problem: How to capture larger sets of periodical granularities?

$\Rightarrow$ we need *more expressive classes of automata*.

Three-phase automata (3PA):

- they recognize languages obtained from Büchi recognizable languages by discarding non ultimately periodic words;
- they operate as follows:
  1. guess the prefix of the word;
  2. guess the repeating pattern and store it in a queue;
  3. recognize the stored pattern infinitely often.

Beyond (R)UPA - 2

Theorem. 3PA-recognizable languages are closed under *union*, *intersection*, *concatenation* with regular languages, and *complementation*.

Remark. Noticeable sets of time granularities are not 3PA-recognizable.

Example. The set of all granularities that group days n by n, that is $\{(\blacksquare^n \blacktriangleleft)^\omega | n \geq 0\}$.

A 3PA that recognizes these repeating patterns must also recognize *all, but finitely many, combinations* of them.

Further Work

Other Open Problems:

- Investigate larger classes of automata:
  - that extend 3PA;
  - that (possibly) preserve closure and decidability properties.

- Temporal logics and automata:
  - temporal logic counterparts of SSA, UPA, and 3PA;
  - a computational framework for pairing temporal logics and automata.