Representing and reasoning with classes and instances of possibly periodic events. Theory, algorithms and applications.

Luca Anselma
Relatore: prof. Paolo Terenzian

Ciclo XVIII - Dottorato di Ricerca in Informatica - Dipartimento di Informatica - Università di Torino
Outline

- Introduction
- Classes+instances
- Periodicity
- Nested periodicity
- Conclusions
Luca Anselma, Representing and reasoning with classes and instances of possibly periodic events

Outline

- Introduction
- Classes+instances
- Periodicity
- Nested periodicity
- Conclusions
Introduction

Time: a ubiquitous problem which deserves specific attention

Important in CS: simulation, databases, workflow management, AI (planning, scheduling, diagnosis, expert systems, NLP, KR)
Introduction

In AI, two mainstreams

- First mainstream: first-order or modal logic formalisms
- Second mainstream: constraint-based approaches; they are focused on dealing with temporal aspects only.

More specific problems \(\Rightarrow\) more efficient reasoning mechanisms.

The approach in the dissertation belongs to the second mainstream.
Introduction

Constraint-based \textit{temporal reasoners}: \textit{application-independent} and \textit{domain-independent} temporal constraints management servers; \textit{knowledge servers} to be coupled with other systems / problem solvers
Introduction

Several types of temporal constraints:
- qualitative temporal constraints (e.g., [Allen, 83])
- quantitative temporal constraints (e.g., [Dechter et al., 91])
- “mixed” qualitative + quantitative (e.g., [Meiri, 91])
- repeated/periodic constraints (e.g., [Morris et al., 93])
- constraints on classes only (e.g., [Morris et al., 93])
- constraints on instances only (e.g., [Allen, 91b])
Real-world examples

From clinical guidelines

One possible therapy for multiple myeloma is made by six cycles of 5-day treatment, each one followed by a delay of 23 days (for a total time of 24 weeks). Within each 5-day cycle, 2 inner cycles can be distinguished: the melphalan treatment, to be provided twice a day, for each of the 5 days, and the prednisone treatment, to be provided once a day, for each of the 5 days. These two treatments must be performed in parallel.

(an excerpt from a clinical guideline for Multiple Myeloma)

The therapy lasts 88 weeks. In the therapy, cotrimoxazole must be given twice daily on two consecutive days every week.

(an excerpt from a clinical guideline for Childhood Acute Lymphoblastic Leukaemia)

Give acetaminophen twice a day until the fever has gone for a maximum of 15 days.
Real-world examples

From clinical guidelines

One possible therapy for multiple myeloma is made by six cycles of 5-day treatment, each one followed by a delay of 23 days (for a total time of 24 weeks). Within each 5-day cycle, 2 inner cycles can be distinguished: the melphalan treatment, to be provided twice a day, for each of the 5 days, and the prednisone treatment, to be provided once a day, for each of the 5 days. These two treatments must be performed in parallel.

(an excerpt from a clinical guideline for Multiple Myeloma)

Melphalan was given to John on Jan 13, 2004 between 12:00 and 12:15 and then from 18:00-18:10 to 18:20.
Prednisone was given to John on Jan 13, 2004 between the two administrations of melphalan…
Issues

- Classes+instances
  - constraint inheritance
  - consistency checking
    (note: underlying semantic assumptions (e.g., future, prediction))

- Composite events

- Repeated/periodic events
  - repetition pattern (nested)
  - conditions

Note: strongly related issues

No integrated approach in the literature
Goals

Approach:
- domain-independent
- tractable
- trade-off
- intensional
Methodology

- problem
- formalism
- algorithms
- properties (trade-off)
- implementation and experiments
- application
Architecture

Layered Approach

High Level: interface (expressive language + reasoning facilities) trade-off (expressiveness vs complexity)

“Low” Level: internal representation + constraint propagation “standard” approach (STP)
Architecture

Layered Approach

**High Level**: interface (expressive language + reasoning facilities)
trade-off (expressiveness vs complexity)

**“Low” Level**: internal representation + constraint propagation
“standard” approach (STP)
Architecture

Layered Approach

High Level: interface (expressive language + reasoning facilities)
trade-off (expressiveness vs complexity)

Filling the GAP: correlation; association; semantic assumptions; inheritance (also periodic patterns); prediction…

“Low” Level: internal representation + constraint propagation
“standard” approach (STP)
Contributions

Three (incremental) approaches:

1. **Classes+instances**
2. Classes+instances + **periodicity**
3. Classes+instances + (**nested** periodicity + **conditions**)

Note. For each approach $\rightarrow$ methodology
Comparisons $\rightarrow$ trade-off analysis
Outline

- Introduction
- Classes+instances
- Periodicity
- Nested periodicity
- Conclusions
Issues

CLASSES

Reservation (RS) 1-7 days before Lab_Test (LT) 1-48 hours before Report (RP)

RS1 RS2 LT1 LT2 RSk LTk RP1 RP2 RPk

INSTANCES

timeline

instance_of

temporal constraints
In order to check the temporal consistency:
- Which actions do inherit constraints? ⇒ correlation
- Semantic assumptions (Complete observation? Future events?)
- Prediction
Low level language

Simple Temporal Problem (STP)

Conjunction of bounds on differences:
\[ l \leq P2 - P1 \leq u \]

\( P1 \) and \( P2 \) are temporal points
\( l \) and \( u \) are numbers

E.g., A lasts 1h to 2h
\[ 1h \leq E_A - S_A \leq 2h \]

B is at least 10 mins after A
\[ 10m \leq S_B - E_A \leq +\infty \]

Consistency checking: constraint propagation algorithm, correct, complete and tractable \( (O(n^3)) \)
ITL (Instances Temporal Language)

before(E1,E2) ⇔
0 < Start(E2) - End(E1)

delay(P1,P2,L,U) ⇔
L ≤ P2 - P1 ≤ U
CTL (Classes Temporal Language)

\[ \text{Cbefore}(C1,C2) \iff (\forall C1',C2',\text{Cor} \ (\text{instanceOf}(C1',C1) \land \text{instanceOf}(C2',C2) \land \text{COR}(\text{Cor},C1',C2'))) \Rightarrow \]
\[ (0 < \text{Start}(C2') - \text{End}(C1')) \land \]
\[ (\forall C1' \text{ instanceOf}(C1',C1) \Rightarrow (\exists C2',\text{Cor} \ \text{instanceOf}(C2',C2) \land \text{COR}(\text{Cor}, C1',C2')))) \]

\[ \text{Cdelay}(\text{Start}(C1),\text{Start}(C2),L1,U1) \iff (\forall C1',C2',\text{Cor} \ (\text{instanceOf}(C1',C1) \land \text{instanceOf}(C2',C2) \land \text{COR}(\text{Cor},C1',C2'))) \Rightarrow \]
\[ (L1 \leq \text{Start}(C2') - \text{Start}(C1') \leq U1) \land \]
\[ (\forall C1' \text{ instanceOf}(C1',C1) \Rightarrow (\exists C2',\text{Cor} \ \text{instanceOf}(C2',C2) \land \text{COR}(\text{Cor}, C1',C2')))) \]

Note. Inheritance, correlation, prediction.
Algorithms

Two orthogonal semantic assumptions: future events/complete observability

No future events:
all instances start before now
**Algorithms**

*Complete observability:*

what happens if a “predicted” instance is not observed within the “predicted” range of time?

- **Incomplete** observations → OK (prediction not needed)
- **Complete** observations → Inconsistency (prediction needed)

Orthogonal assumptions → 4 consistency checking algorithms
Algorithms

E.g.: Future events/no complete observability

function Integrated_Reasoning_IF(CKB = <CKB_Elements, STP_CKB_Con>, IKB = <IKB_Elements, IKB_instanceOf, IKB_COR, STP_IKB_Con>) : STP
1. pre-compute the pairs of connected nodes in CKB
2. STP_IKB_Con' ← FloydWarshall(STP_IKB_Con)
3. STP_CKB_Con' ← FloydWarshall(STP_CKB_Con)
4. IKB_COR ← Closure(IKB_COR)
5. check consistency of correlations
6. Temporal_Constraint_Inheritance(<CKB_Elements, STP_CKB_Con'>, <IKB_Elements, IKB_instanceOf, IKB_COR, STP_IKB_Con'>)
7. Minimal_Network ← FloydWarshall(STP_IKB_Con')
8. return Minimal_Network
procedures Temporal_Constraint_Inheritance(CKB = <CKB_Elements,STP_CKB_Con'>, IKB = <IKB_Elements, IKB_instanceOf, IKB_COR, STP_IKB_Con'>)

1. forall E ∈ IKB_Elements do
   let C ∈ CKB_Elements the corresponding class /* i.e.,
   instanceOf(E,C) holds */
   let t ≤ End(C) - Start(C) ≤ u ∈ STP_CKB_Con' the constraint
   on the duration of C (if any)
   STP_IKB_Con' ← STP_IKB_Con' ∪ \{t ≤ End(E) - Start(E) ≤ u\}
   let t1 ≤ Start(C) – X0 ≤ u1 ∈ STP_CKB_Con' and t2 ≤ End(C)
   – X0 ≤ u2 ∈ STP_CKB_Con' the constraints on the date of C (if any)
   STP_IKB_Con' ← STP_IKB_Con' ∪ \{t1 ≤ Start(E) – X0 ≤ u1, t2 ≤ End(E) – X0 ≤ u2\}; od

2. forall E1,E2 ∈ IKB_Elements, E1≠E2\ Exists Cor COR(Cor,E1,E2) do
   let C1 ∈ CKB_Elements and C2 ∈ CKB_Elements the
   corresponding classes /* i.e., instanceOf(E1,C1) and
   instanceOf(E2,C2) hold */
   instantiate on E1 and E2 the constraints in CKB_Constraints
   between C1 and C2; od
Properties

Properties 3.1, 3.2, 3.3:
The algorithms are correct and complete

Complexity:
The algorithms are tractable
Outline

- Introduction
- Classes+instances
- Periodicity
- Nested periodicity
- Conclusions
Issues

Which events do inherit constraints? \( \Rightarrow \) association to periodicity

E.g.: “twice each day A (before B, ... C, ... D)"

Inheritance: periodic patterns (n-ary constraints!)
$CTL_P$ (Classes Temporal Language)

$CTL +$ repetition/periodicity constraint:

$\text{repetition}(C, FT, AT, DT, IT, \text{freq})$
CTL_p (Classes Temporal Language)

Ex. 4.4. For six months, perform action A twice each five days for twenty days and then suspend for ten days (and so on).

repetition(A, 180d, 20d, 10d, 5d, 2)
CTL\textsubscript{P} (Classes Temporal Language)

\begin{align*}
\text{repetition}(C, FT, AT, DT, IT, freq) &\iff \\
\forall C' (\text{Instance}_\text{of}(C', C) \Rightarrow (\exists FT, AT+DT_1, \ldots, AT+DT_{nATDT}, AT_1, \ldots, AT_{nATDT}, DT_1, \ldots, DT_{nATDT}, IT_{1,1}, \ldots, IT_{nATDT,nIT}, freq_{1,1,1}, \ldots, freq_{nATDT,nIT,freq} \quad C_{1,1,1}, \ldots, C_{nATDT,nIT,freq})(\\
\% durations % \\
C_{\text{duration}}(FT, FT, FT) & \land \\
\forall i=1, \ldots, nATDT (C_{\text{duration}}(AT+DT_i, AT+DT, AT+DT) & \land C_{\text{duration}}(AT_i, AT, AT) \\
& \land C_{\text{duration}}(DT_i, DT, DT) & \land \forall j=1, \ldots, nIT (C_{\text{duration}}(IT_{ij}, IT, IT))) & \land \\
\% containment % \\
\forall i=1, \ldots, nATDT (C_{\text{containment}}(AT+DT_i, FT) & \land C_{\text{containment}}(AT_i, AT, AT) \\
& \land C_{\text{containment}}(DT_i, DT, DT) & \land \forall j=1, \ldots, nIT (C_{\text{containment}}(IT_{ij}, IT, IT))) & \land \\
\% anchor to the start % \\
C_{\text{starts}}(AT+DT_i, FT) & \land \\
\forall i=1, \ldots, nATDT (C_{\text{starts}}(AT_i, AT+DT_i) & \land C_{\text{starts}}(IT_{ij}, AT_i) & \land \\
& \land \forall j=1, \ldots, nIT (C_{\text{starts}}(freq_{ij,k}, IT_{ij})) & \land C_{\text{starts}}(C_{ij,k, freq_{ij,k}})) & \land \\
\% ordering and covering % \\
\forall i=1, \ldots, nATDT-1 (C_{\text{meets}}(AT+DT_i, AT+DT_{i+1}) & \land C_{\text{meets}}(AT_i, DT_i) & \land \\
& \land \forall j=1, \ldots, nIT-1 (C_{\text{meets}}(IT_{ij}, IT_{ij+1}) & \land \\
& \land \forall k=1, \ldots, freq-1 (C_{\text{meets}}(freq_{ij,k}, freq_{ij,k+1}))) & \land \\
\% instances % \\
\forall i=1, \ldots, nATDT, j=1, \ldots, nIT, k=1, \ldots, \text{freq (Instance}_\text{of}(C', C_{ij,k}))))
\end{align*}
Low level

Goal: *intensional* representation

STP is not enough!

Extension to STP:

STP-Tree
“One possible therapy for multiple myeloma is made by six cycles of 5-day treatment, each one followed by a delay of 23 days (for a total time of 24 weeks). Within each 5-day cycle, 2 inner cycles to be performed in parallel can be distinguished: the melphalan treatment, to be provided twice a day, for each of the 5 days, and the prednisone treatment, to be provided once a day, for each of the 5 days.”
function integratedConsistency_p(T : CKB, I : IKB) : STP
1. S ← visitSTPTreeP(T)
2. add to I the placeholder instances corresponding to the placeholder classes in S
3. add to I the placeholder instances corresponding to the non-atomic, non-repeated classes in S
4. I' ← FloydWarshall(I)
5. if I' = INCONSISTENT then return INCONSISTENT
6. for each class C in S in temporal order do
7. let i be the first instance of C in I' not yet taken in consideration
8. if i does not exist then return INCONSISTENT
   od
9. if there exists an instance in I' not yet considered then return INCONSISTENT
10. Temporal_Consstraint_Inheritance(T, I')
11. I'' ← FloydWarshall(I')
12. if I'' = INCONSISTENT then return INCONSISTENT else return I''
Properties

Properties 4.2, 4.3:
The algorithms are *correct* and *complete*

Complexity:
The algorithms are *tractable*

Note: assumption of total ordering among instances.
Luca Anselma, Representing and reasoning with classes and instances of possibly periodic events

Outline

- Introduction
- Classes+instances
- Periodicity
- Nested periodicity
- Conclusions
Issues

The therapy lasts 88 weeks. In the therapy, cotrimoxazole must be given twice daily on two consecutive days every week.

Give acetaminophen twice a day until the fever has gone for a maximum of 15 days.

The repetitions are not only periodic, but also: multiply nested (2 cotrimoxazole(s) ⊆ a day ⊆ a week ⊆ 88 weeks), and imprecisely located (in what days of the week do they occur?).

Also temporal constraints between repetitions are needed (“consecutive days”)

and conditioned repetitions (“until the fever has gone”).
CTL$_{RP}$ (Classes Temporal Language)

CTL + repetition/periodicity constraint

Syntax (BNF):
Rep ::= repetition$_{RP}$(A, RSpec)
A ::= class
RSpec ::= [Nrep, ITime, {RConstr}, {Cond}] | [Nrep, ITime, {RConstr}, {Cond}], RSpec
Nrep ::= natural number
ITime ::= exact duration
RConstr ::= {fromStart(m,m)} {Bet} {toEnd(m,m)}
Bet ::= inBetweenAll(m,m) | inBetween(m,m)
m ::= duration
Cond ::= {while(C)} | {onlyIf(C)}
C ::= boolean expression

Semantics:
Extensional based on the components
Properties

Property 5.3: The algorithms are correct and complete

Complexity: The algorithms are tractable

Note: assumption of “belonging” of instances to classes.
Outline

- Introduction
- Classes+instances
- Periodicity
- Nested periodicity
- Conclusions
Expressiveness

Property 5.1.
\( \text{CTL}_{RP} \) subsumes \( \text{CTL}_P \) and \( \text{CTL}_P \) subsumes \( \text{CTL} \).
### Summary

<table>
<thead>
<tr>
<th>Periodic events</th>
<th>Complexity of consistency checking</th>
<th>Assumptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CTL</strong></td>
<td>No</td>
<td>O(max{C^3,I^3})</td>
</tr>
<tr>
<td><strong>CTL_p</strong></td>
<td>Yes (not nested)</td>
<td>O(max{C^3,I^3})</td>
</tr>
<tr>
<td><strong>CTL_Rp</strong></td>
<td>Yes (nested and conditioned)</td>
<td>O(max{C^3,I^3})</td>
</tr>
</tbody>
</table>
## Summary

<table>
<thead>
<tr>
<th>Periodic events</th>
<th>Complexity of consistency checking</th>
<th>Assumptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>$O(\max{C^3, t^3})$</td>
<td>None</td>
</tr>
<tr>
<td>Yes (not nested)</td>
<td>$O(\max{C^3, t^3})$</td>
<td>Total ordering</td>
</tr>
<tr>
<td>Yes (nested and conditioned)</td>
<td>$O(\max{C^3, t^3})$</td>
<td>Belonging</td>
</tr>
</tbody>
</table>
### Summary

<table>
<thead>
<tr>
<th>Periodic events</th>
<th>Complexity of consistency checking</th>
<th>Assumptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CTL</strong></td>
<td>No</td>
<td>$O(\max{C^3, I^3})$</td>
</tr>
<tr>
<td><strong>CTL_p</strong></td>
<td>Yes (not nested)</td>
<td>$O(\max{C^3, I^3})$</td>
</tr>
<tr>
<td><strong>CTL_{RP}</strong></td>
<td>Yes (nested and conditioned)</td>
<td>$O(\max{C^3, I^3})$</td>
</tr>
</tbody>
</table>
## Summary

<table>
<thead>
<tr>
<th>Periodic events</th>
<th>Complexity of consistency checking</th>
<th>Assumptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>$O(\max{C^3, I^3})$</td>
<td>None</td>
</tr>
<tr>
<td>Yes (not nested)</td>
<td>$O(\max{C^3, I^3})$</td>
<td>Total ordering</td>
</tr>
<tr>
<td>Yes (nested and conditioned)</td>
<td>$O(\max{C^3, I^3})$</td>
<td>Belonging</td>
</tr>
</tbody>
</table>

Luca Anselma, Representing and reasoning with classes and instances of possibly periodic events
Luca Anselma, Representing and reasoning with classes and instances of possibly periodic events

**Implementation**

Loosely coupled architecture

Experimental evaluation
Application

Temporal Server

High Level

Low Level

Interface

consistency-checking-guideline
consistency-checking-instance
next-action query
yes/no query
extract query
hypothetical query
temporal simulation

Guideline System
Related works

- Morris et al.’s [93, 96] recurrence algebra
- [Morris et al., 95] generating a consistent scenario
- [Cukierman & Delgrande, 98, 00] periodicity and composite temporal objects
Future Works

Tractable consistency checking ⇒
- limitations in the expressive power (e.g., no disjunctions “before or after”);
- assumptions (regarding instances of periodic events).

Possible extensions: tractable approach as the “core” of a more general (and not tractable) approach:
- removing assumptions;
- dealing with (different forms of) disjunctions.

Application to conceptual database design.

Luca Anselma, Representing and reasoning with classes and instances of possibly periodic events
The end.

Thanks for your attention...
Additional slides...
Problem

MANY REAL-WORLD APPLICATIONS NEED MORE:
- integration
- extensions

* qualitative + quantitative + repetition/periodicity

* classes + instances

* composite events (part-of)
Constraints about “classes”

E.g.: clinical guidelines

One possible therapy for multiple myeloma is made by six cycles of 5-day treatment, each one followed by a delay of 23 days (for a total time of 24 weeks). Within each 5-day cycle, 2 inner cycles can be distinguished: the melphalan treatment, to be provided twice a day, for each of the 5 days, and the prednisone treatment, to be provided once a day, for each of the 5 days. These two treatments must be performed in parallel.

.... plus (temporal) constraints between the therapy and other actions (e.g., diagnosis) in the guideline
⇒ classes
⇒ periodicity/repetition
⇒ composite actions (part-of)

* PROBLEM: intensional vs extensional temporal reasoning (1)
Constraints about classes + instances + composite + repeated/periodic

- Semantic assumptions (complete observations)

Only “atomic” actions are observed

E.g., “A twice a day”. “A: A1 followed by A2”
Issues

But constraints about composite actions have to be inherited/checked!
E.g., “A twice a day”. “A: A1 followed by A2".
“A ends at least 1h before start B”

* PROBLEM: intensional vs extensional temporal reasoning (2)
Goal of the approach

Knowledge server for (constraint-based) temporal reasoning

Let users abstract from these (general, domain-independent) problems

⇒ solve these problems once and for-all

⇒ “hide” complexity
Outline

- Introduction
- Problems
- Strategy
- Solutions (hint)
- Conclusions
Simple Temporal Problem (STP)

- **Temporal network** (directed weighted graph)
  - Time points $\rightarrow$ nodes
  - Constraint $P_2 - P_1 \leq u$ $\rightarrow$
    edge from $P_1$ to $P_2$ with weight $u$

- **All-pairs shortest paths algorithms** (e.g., Floyd-Warshall’s) propagate constraints:
  - check consistency (i.e., detect negative cycles)
  - give minimal network
  - complexity $O(|V|^3)$
Problem

FILLING THE GAP!!!

HIGH LEVEL: interface (expressive language + reasoning facilities)

NOT JUST “MAPPING”: algo required for coping with
- correlation
- association
- semantic assumptions
- inheritance (also periodic patterns)
- prediction

“LOW” LEVEL: internal representation + constraint propagation (STP)
Digression

Idea: STP as a well-established assembly language

APPROACH similar to:
- defining a high-level (programming) language
- designing an interpreter/compiler based on a given assembly language
Outline

- Introduction
- Problems
- Strategy
- Solutions (hint)
- Conclusions
HLL Language (Classes)

CKB = <Classes, Part-of, Constraints>

(1) list of classes (e.g., {Reservation, Lab_Test, ….})
(2) list of “part-of” relations

Constraints (ex.1): delay

Cdelay(Start(C1), Start(C2), L, U) ⇔
(∀ C1', C2' (Instance_of(C1', C1) ∧ Instance_of(C2', C2) ∧ COR(C1', C2')) ⇒
(Start(C2') − Start(C1') ≤ U) ∧
(∀ C1' Instance_of(C1', C1) ⇒ (∃ C2' Instance_of(C2', C2) ∧ COR(C1', C2'))))
Periodicity constraints:

\[ \text{Repetition}(E, \text{RepSpec}) \]

where \( \text{RepSpec} \) is a recursive structure.

Each level:

\[ R_i ::= <n_{\text{Rep}_i}, I-\text{Time}_i, \text{constrs}_i, \text{cond}_i> \]

i.e., \( R_i \) means:

"repeat \( n_{\text{Rep}_i} \) times \( R_{i+1} \) in a time lapse lasting \( I-\text{Time}_i \) following the constraints \( \text{constrs}_i, \text{if cond}_i \) holds"
Periodicity constraints:

\[
\text{Repetition}(E, \langle R_1, \ldots, R_n \rangle)
\]

\[
R_i := \langle n_{\text{Rep}_i}, I\text{-Time}_i, \text{constrs}_{i}, \text{cond}_i \rangle
\]

\text{constrs}_i \text{ imposes temporal constraints between repetitions. It is a (possibly empty) set of:}

- \text{inBetweenAll}(\text{min}, \text{max});
- \text{inBetween}(\langle \text{min}_1, \text{max}_1 \rangle, \ldots, \langle \text{min}_{n_{\text{Rep}-1}}, \text{max}_{n_{\text{Rep}-1}} \rangle);
- \text{fromStart}(\text{min}, \text{max});
- \text{toEnd}(\text{min}, \text{max}).
Periodicity constraints:

\[
\text{Repetition}(E, \langle R_1, \ldots, R_n \rangle) \\
R_i := \langle nRepi, I\text{-Time}_i, constrs_i, \text{cond}_i \rangle
\]

\text{cond}_i \text{ represents conditions that influence the repetitions. It is a (possibly empty) set of:}

- \text{while}(C);
- \text{onlyIf}(C).
Example:

“Intrathecal methotrexate must be administered 7 times during 88 weeks, never less than 10 weeks apart or more than 14 weeks apart”.

(an excerpt from a clinical guideline for Childhood Acute Lymphoblastic Leukaemia)

Repetition(IntrathecalMethotrexate, <<7,88wk,{inBetweenAll(10wk,14wk)}>>
Example:

“The therapy lasts 88 weeks. In the therapy, cotrimoxazole must be given twice daily on two consecutive days every week”.

(an excerpt from a clinical guideline for Childhood Acute Lymphoblastic Leukaemia)

Repetition (cotrimoxazole,
<<88, 88wk, {}>,
<2, 1wk, {inBetweenAll(0,0)}>,
<2, 1d, {}}>)
HLL Language (Instances)

IKB \equiv \langle 1), (2), (3), (4)\rangle

(1) list of instances (e.g., \{RS1, RS2, LT1, ….\})

(2) INSTANCE-OF (e.g., Instance_of(RS1, Reservation))

(3) CORRELATION (e.g., COR(RS1,LT1))

(4) CONSTRAINTS (examples):
\text{delay}(P1,P2,L1,U1) \iff L1 \leq P2 - P1 \leq U1
Reasoning Mechanisms
Checking consistency on classes

**procedure** STPTreeConsistency(X: STPNode, RepSpec)

1) let Max the maximum duration of a repetition
2) impose (in X) that the maximum distance between each pair of points is the minimum between the current maximum distance and Max
3) X ← FloydWarshall(X)
4) **return** X

**Complexity.** STPTreeConsistency runs in \( O(\max\{L, C^3\}) \), where \( L \) is the maximum number of nesting levels of the repetitions and \( C \) is the number of classes.
Reasoning Mechanisms
Checking consistency on classes

procedure STP_tree_consistency(X: STPNode, RepSpec)
1) let RepSpec = \{ R_1 = \{ nRep_1, IT_1, constrs_1 \},
                      \ldots, R_n = \{ nRep_n, IT_n, constrs_n \} \}
2) for i = 1 to n do
3)   max_i \leftarrow ( IT_i - ( \text{fromStart}_i \cdot \text{min} + 
                      \sum_{j \in \text{between}_i[j]} \cdot \text{min} + \text{toEnd}_i \cdot \text{min} )) / nRep_i
4)   if i < n then
5)     if IT_{i+1} > max_i then return INCONSISTENT
6)     else
7)        impose (in X) that the maximum distance
            between each pair of points is the
            minimum
            between the current maximum distance and
            max_n
8)    od
9)    X \leftarrow \text{FloydWarshall}(X)
10) if X = INCONSISTENT then return INCONSISTENT
    else return X
Reasoning Mechanisms
Checking consistency on classes+instances

procedure integratedConsistency(T : CKB, I : IKB)
1) associate the classes with the instances
2) inherit (in I) the constraints from classes to instances
3) add to I the constraints related with the repetition
   constraints
4) $I \leftarrow \text{FloydWarshall}(I)$
5) return $I$

Complexity. integratedConsistency runs in $O(\max\{I^3, C^3\})$, where $I$ is the number of instances.
(assumptions: complete observability in the past and total temporal ordering of instances).
Reasoning Mechanisms
Checking consistency on classes+instances

procedure integratedConsistency(T : CKB, I : IKB, NOW)
  1) add to I the constraints that all instances start before NOW
  2) unfoldNode(Root(T), I)
  3) if there exists an instance in I not yet considered then return INCONSISTENT
  4) I := FloydWarshall(I)
  5) if I=INCONSISTENT then return INCONSISTENT
  6) for each instance i not found by unfoldNode do
     7) if NEC(Start(i) before NOW) then
     8) return INCONSISTENT
Reasoning Mechanisms

procedure unfoldNode(X : STPNode, I : IKB)
1) add to I the placeholder class $C_X$
2) forall $C_A$ | $C_A$ is not a repeated class in X do
3) find in I the first instance $i$ corresponding with class $C_A$
4) if $i$ does not exist then add it to I
5) od
6) forall $C_R$ | $C_R$ is a repeated class in X do
7) let RepSpec = $<R_1, \ldots, R_n>$ be the repetition specification of class $C_R$
8) $C_{sub} :=$ unfoldRep($X$, I, $C_R$, RepSpec)
9) add to I the constraints that $C_{sub} \subseteq C_X$
10) od
11) for each monadic constraint in X do
12) add the constraint to the corresponding classes in I
13) for each binary constraint in X do
14) add the constraint to the corresponding classes in I
15) return $C_X$
Reasoning Mechanisms

procedure unfoldRep(X : STPNode, I : IKB, C : Class, RepSpec = <R₁, R₂,..., Rₙ>)
1) add to I the placeholder class C₁
2) let R₁ = <nRep₁, IT₁, constrs₁>
3) for r := 1 to nRep₁ do
   4) if R₁ is not the last level in RepSpec then
      5)   Csubᵣ := unfoldRep(X, I, C, <R₂, ..., Rₙ>)
      6) else
         7)   Csubᵣ := unfoldNode(child(C, X), I)
   8) od
9) add to I the constraint that duration of C₁ is IT₁
10) add to I the constraints that Csubᵢ ⊆ C₁
11) add to I the constraints that Csubᵢ₊₁ is after Csubᵢ
12) add to I the constraint fromStart in constrs₁ in R₁ between C₁ and Csubᵢ
13) add to I the constraint toEnd in constrs₁ in R₁ between Csubᵣ and C₁
14) add to I the constraints inBetween and inBetweenAll in constrs₁ in R₁ between Csubᵢ and Csubᵢ₊₁
15) return C₁
## Reasoning Mechanisms

<table>
<thead>
<tr>
<th></th>
<th>Consistency checking on classes</th>
<th>Integrated consistency checking</th>
<th>Periodic events</th>
<th>Observability</th>
<th>Supported events</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>X</td>
<td>X</td>
<td>–</td>
<td>complete / not complete</td>
<td>past events / all events</td>
</tr>
<tr>
<td>II</td>
<td>X</td>
<td>X</td>
<td>X (not recursive)</td>
<td>complete</td>
<td>past events / all events</td>
</tr>
<tr>
<td>III</td>
<td>X</td>
<td>X</td>
<td>X (recursive and conditioned repetitions)</td>
<td>complete</td>
<td>past events / all events</td>
</tr>
</tbody>
</table>

Complexity of consistency checking on classes: $O(C^3)$
Complexity of integrated consistency checking: $O(\max\{C^3, I^3\})$
Temporal facilities

Based on the algorithms:

- **consistency checking of classes**
- **consistency checking of classes+instances**
- **query** to obtain temporal information:
  - *next-action* query (when the next actions have to be performed);
  - *yes/no* query (ask whether a given set Q of temporal constraints is possible given KB (i.e., if it is consistent with KB)).
  - *extract* query (outputs the minimal and maximal distance between a pair of actions);
  - *hypothetical* query (ask queries in the hypothesis that a further set TC of temporal constraints is assumed);
- **temporal simulation** (see the temporal consequences of choosing among different alternative paths – “what if?”)
Outline

- Introduction
- Problems
- Strategy
- Solutions (hint)
- Conclusions
Conclusions

General-purpose and domain-independent knowledge server which supports:
- qualitative and quantitative temporal constraints;
- classes and instances of events;
- composite events;
- periodic events.

(Sample) application areas: planning, workflow management, protocol/guidelines management

Applied to clinical guidelines
(San Giovanni Battista, Torino, Italy; Cancer Research UK)
procedure path-consistency

Until no change is made

forall triples of variables $x_i$, $x_j$, $x_k$

$$C_{ik} \leftarrow C_{ik} \cap (C_{ij} \odot C_{jk})$$
CTL_{RP} (Classes Temporal Language)

Repetition(C, <R=<nRep, IT, constrs>>) ⇔

(∃ E, E_1, …, E_{nRep} (  
% duration %  
Cduration(E, IT, IT) ∧  
% containment %  
∀ i=1, …, nRep (CNSduring(E_i, E) ∧  
% ordering %  
∀ i=1, …, nRep-1 (CAfter(E_{i+1}, E_i)) ∧  
% constraints in constrs must hold % … ∧  
% instances %  
(∃ I_1, …, I_{nRep} (Instance_of(I_i, C) ∧ AllDifferent(I_1, …, I_{nRep}))))

Luca Anselma, Representing and reasoning with classes and instances of possibly periodic events
CTL_{RP} (Classes Temporal Language)

...% constrs %

\((\text{inBetweenAll}(\text{minIBA}, \text{maxIBA}) \in \text{constrs} \Rightarrow (\forall i=1, \ldots, n\text{Rep} \ (\text{CDelay(End(E_i), Start(E_{i+1}), minIBA, maxIBA))})) \land

(\text{inBetween}((\text{minIB}_1, \text{maxIB}_1), \ldots, (\text{minIB}_{n\text{Rep}-1}, \text{maxIB}_{n\text{Rep}-1}) ) \in \text{constrs} \Rightarrow (\text{CDelay(End(E_1), Start(E_1), minIB}_1, \text{maxIB}_1) \land \ldots \land \text{CDelay(End(E}_{n\text{Rep}-1), \text{Start(E}_{n\text{Rep}-1}, \text{minIB}_{n\text{Rep}-1}, \text{maxIB}_{n\text{Rep}-1}) )) ) \land

(\text{fromStart}(\text{minS}, \text{maxS}) \in \text{constrs} \Rightarrow \text{CDelay(Start(E), Start(E_1), minS, maxS)}) \land

(\text{toEnd}(\text{minE}, \text{maxE}) \in \text{constrs} \Rightarrow \text{CDelay(End(E}_{n\text{Rep}), \text{End(E)}, \text{minE}, \text{maxE})}) \land

...
procedure FloydWarshall
forall triples of variables $x_i$, $x_j$, $x_k$
if $\text{dist}[i,j] > \text{dist}[i,k] + \text{dist}[k,j]$
\[ \text{dist}[i,j] = \text{dist}[i,k] + \text{dist}[k,j] \]