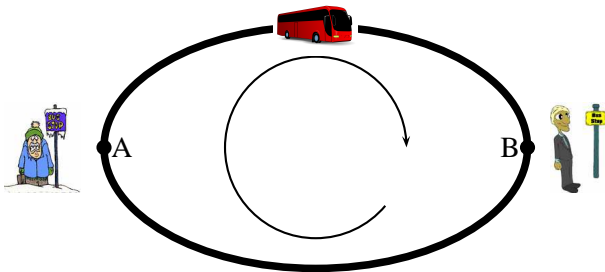


*REPRESENTATION OF
PERIODIC MOVING OBJECTS IN DATABASES*

Motivation

- some existing database systems (e.g., Secondo) can store moving objects
- movement is stored as a sequence of simple movements
- if a movement is repeated n times, this movement is also stored n times



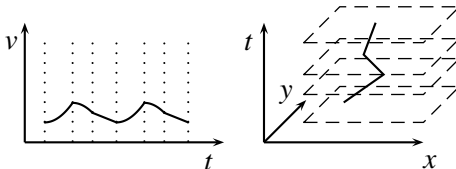
AB BB BA AA AB BB BA AA AB BB BA AA AB

Goals

- avoid multiple storing of repeated movement
changes which are repeated several times should be stored only once
- nested repetitions
the model should cover also more complicated types of repetitions
- representation of non-periodic movements
the model is not restricted to periodic movements
- independency of the underlying type
the model should work with different data types in the same way

Basic Ideas

- sliced representation



divide the object def. time into slices, each slice describes a simple movement

- relative intervals

instead of absolute intervals, use relative intervals (duration + closure properties)

- tree representation

represent a moving object by a tree with different kinds of nodes

Representation of Time: Relative Intervals

- New data types:
 - duration:
 - A value is a distance between two instant values
 - $D_{\text{duration}} = \{d \in \text{real} \wedge d \geq 0\}$
 - relinterval:
 - A value is a relative interval
 - $D_{\text{relinterval}} = \{(l, lc, rc) \mid l \in D_{\text{duration}}, lc, rc \in \text{bool}\}$
 - if $l = 0$, then $lc = rc = \text{TRUE}$
- Anchor:
 - $((l, lc, rc) \leftarrow t) = (t, t + l, lc, rc)$,
where $(t, t + l, lc, rc) \in \text{Interval}(D_{\text{instant}})$, and $t \in D_{\text{instant}}$

Periodic Temporal Types: Units

- New definition of units:
 - $Unit(S_\alpha) = D_{\text{relinterval}} \times S_\alpha$, where $\alpha \in \{pmpoint, pmreal, pmbool, \dots\}$
- New temporal evaluation function:
 - $\iota_\alpha : S_\alpha \times D_{\text{duration}} \rightarrow D_\alpha$

Example Unit: Periodic Moving Point

- **Containment:**

$$p \in (l, rc, lc) \Leftrightarrow \forall t \in \overline{D}_{\text{instant}}((t + p) \in (t, t + l, rc, lc))$$

- $D_{\text{upoint}} = D_{\text{reinterval}} \times D_{\text{point}} \times D_{\text{point}}$
- For $(i, (x_1, y_1), (x_2, y_2)) \in D_{\text{upoint}}$ and $p \in D_{\text{duration}}$:

$$\iota_{\text{pmpoint}}((i, (x_1, y_1), (x_2, y_2)), p) = \begin{cases} (x_r, y_r) & \text{if } p \in i \\ \perp & \text{otherwise} \end{cases}$$

where

$$x_r = x_1 + \text{fraction}(i, p) * (x_2 - x_1)$$

$$y_r = y_1 + \text{fraction}(i, p) * (y_2 - y_1)$$

- **Fraction:**

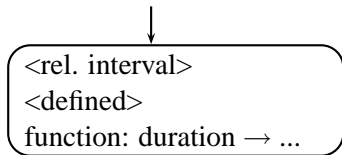
$$\text{fraction}((l, lc, rc), p) := \begin{cases} \frac{p}{l} & \text{if } 0 < p < l \wedge l \neq 0 \\ 1 & \text{if } p = l \wedge rc \wedge l \neq 0 \\ 0 & \text{if } p = 0 \wedge lc \\ \perp & \text{otherwise} \end{cases}$$

Periodic Temporal Types: The Tree Representation

- Basic movement nodes
- Composite movement nodes
- Periodic movement nodes

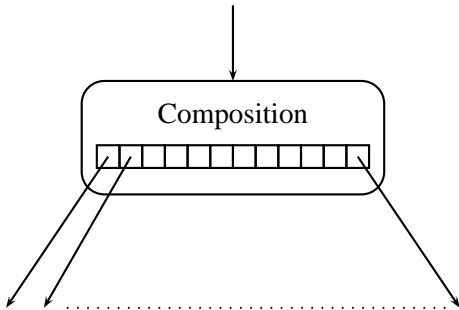
Basic Movement Nodes

- corresponds to a single slice (unit)
- leaves of the tree



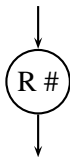
Composite Movement Nodes

- describe a linear sequence of submovements
- minimum 2 submovements have to exist
- holes in the definition time modeled by undefined basic movements



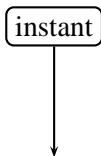
Periodic Movement Nodes

- represents a repeated submovement
- has exactly one submovement which must not be a periodic one
- stores the number of repetitions



The Root of the Tree

- all non-root nodes use relative intervals
- the root stores an anchor time as starting instant of the represented movement



Example

A

B

C

Example

(A)

(B)

starts at 8:00 AM from A

(C)

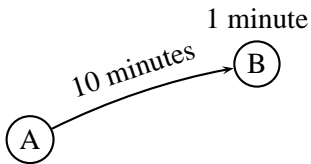
Example



starts at 8:00 AM from A



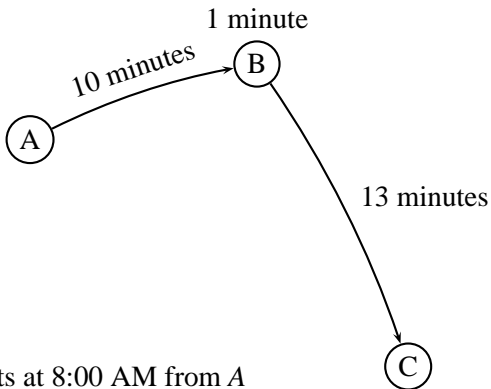
Example



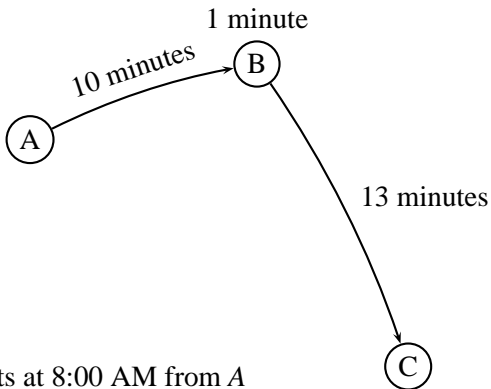
starts at 8:00 AM from A



Example



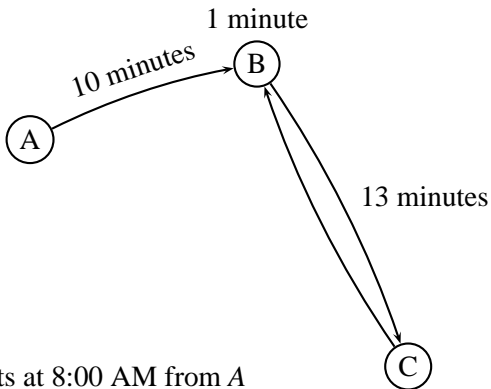
Example



starts at 8:00 AM from A

1 minute

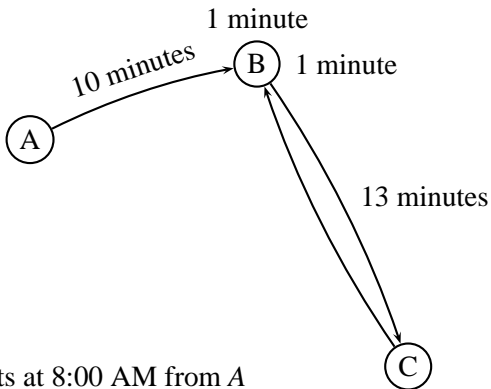
Example



starts at 8:00 AM from A

1 minute

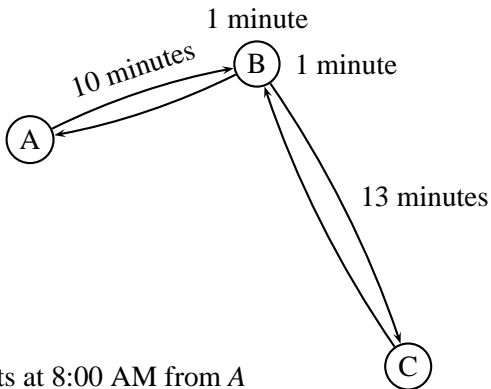
Example



starts at 8:00 AM from A

1 minute

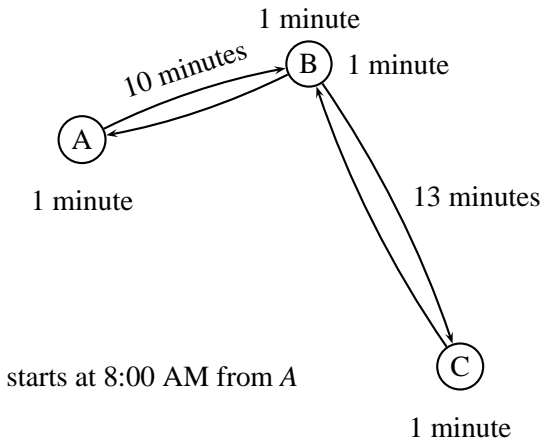
Example



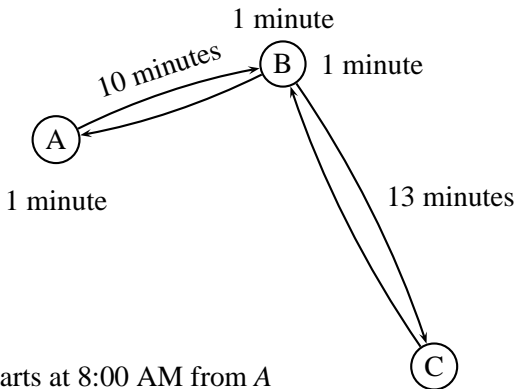
starts at 8:00 AM from A

1 minute

Example



Example

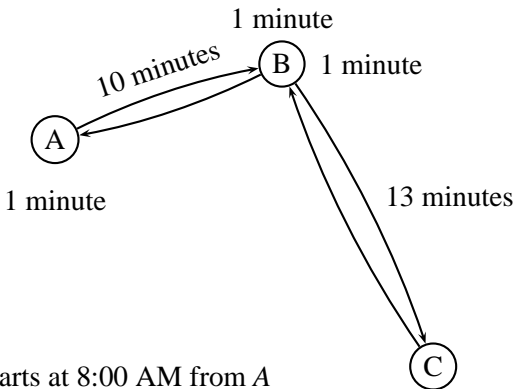


starts at 8:00 AM from A

repeated until 4:20 PM

1 minute

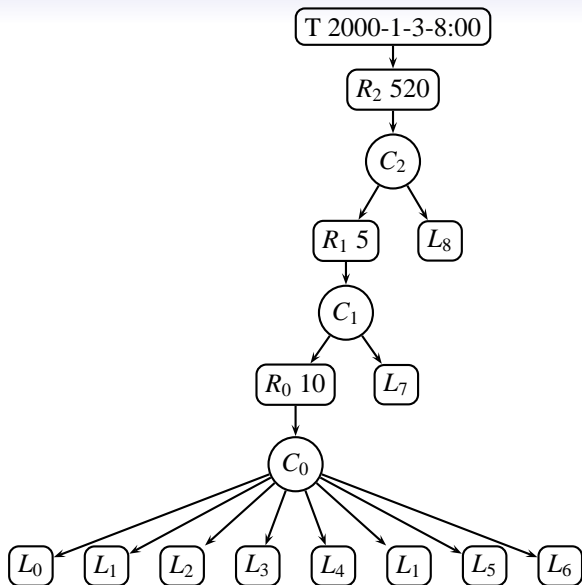
Example



starts at 8:00 AM from A

repeated until 4:20 PM

remaining day and at the weekend the train stays at station A



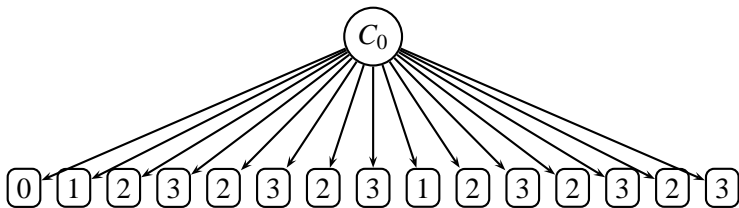
- C_0 : a single run
- L_7 : staying at the night
- C_1 : a single working day
- L_8 : staying on weekend
- C_2 : a complete week
- R_2 : lifetime of the train

Conversion from the Flat to the Tree Representation: Detection of Repetitions in Movements

- problem: detection of (nested) repetitions in movements
- starts with a single composite movement node
- algorithm works bottom up
- searches for repetitions of growing length

Detection of Repetitions in Movements-Example

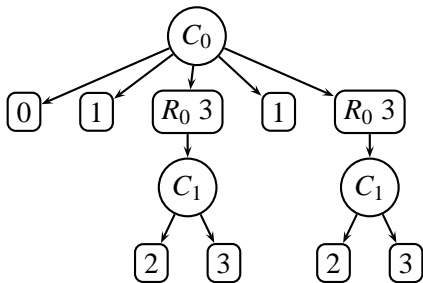
- analyse sequence 0 1 2 3 2 3 2 3 1 2 3 2 3 2 3
- each number represents an ID of a basic movement (unit)



- construct a single composition
- no repetition of length 1
- repetition of length 2 (232323 and 232323)

Detection of Repetitions in Movements-Example

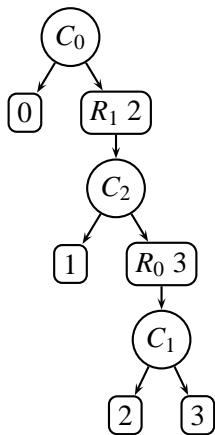
resulting tree is:



- repetition of length 2 (1R0)

Detection of Repetitions in Movements-Example

resulting tree



no further repetitions

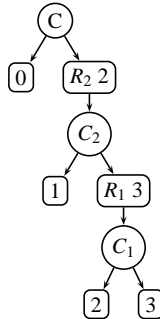
Some Operations

atinstant computes the state of a periodic moving object at a given point in time

traverse the tree from top to a leaf, perform a binary search on composite nodes

trajectory compute the projection of a periodic moving point onto the space
scan only the leaves to construct the result

distance computes the distance to a non-moving point
copy the structure, change the content of the leaves

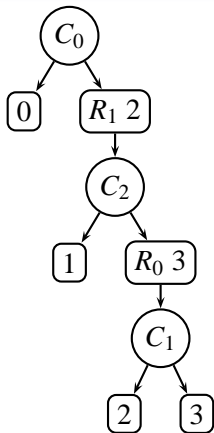


Secondo Implementation of the Tree

A *root record* that has 7 fields:

- **anchor**
start of the whole movement (time instant)
- **duration**
the duration of the whole movement
- **topmove**
(DB array ID, DB array index)
- **DB array** of basic movements
an element is a unit
- **DB array** of periodic movements
an element is: (DB array ID, DB array index, # of repetitions, duration)
- **DB array** of composite movements
an element is: (DB array index, DB array index, duration)
- **DB array** of composite submovements
an element is: (DB array ID, DB array index, duration of the left siblings)

Secondo Implementation of the Tree: Example



anchor (C, 0)

	0	1	2	3
B	U_0	U_1	U_2	U_3

	R_0	R_1
P	(C, 0) 3	(C, 1) 2

	C_0	C_1	C_2
C	(0, 1)	(2, 3)	(4, 5)

S	(B, 0)	(P, 1)	(B, 1)	(P, 0)	(B, 2)	(B, 3)
----------	--------	--------	--------	--------	--------	--------

Comparison with the flat model

- “flat model”
 - uses slices with anchored intervals
 - holes are not represented
 - repetitions are not exploited
 - implemented as another algebra module within SECONDO
- two datasets
 - generated data:
underground trains of Berlin (the best case)
 - collected data:
traces from a GPS receiver without any repetition
(worst case)
- comparison criteria
 - storage size
 - used time for the *trajectory* and *atinstant* operation

Results of the Comparison

Trains

	flat model	periodic model
storage size	429.56 MB, 4 M units	0.37 MB, 3 K basic nodes
<i>trajectory</i> (s)	149	0.25
<i>atinstant</i> (s)	6.7	0.0033

GPS-Traces

	flat model	periodic model
storage size	35.58 MB, 300 K units	39.43 MB, 300 K basic nodes
<i>trajectory</i> (s)	11.4	11.2
<i>atinstant</i> (s)	0.58	0.33