Axiomatic semantics of time and causality models in TESL

Preliminary Research Report

July 17, 2015

Contents

1 Mathematical background

We give hereafter some useful mathematical notations for the reader and shall assume that she or he is familiar to set and recursion theory. That is, $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{D}$ will denote usual sets in mathematics. In particular, $\mathbb{U} = \{\star\}$ is the unit set which contains exactly one element and $\mathbb{F}$ is the set of floating-point numbers as defined in IEEE 754.

1.1 Sets, products and Kleene stars

Let S be a set. We denote $\mathcal{P}_f(S)$ as the finite superset of S and yet S^n as the n-time *cartesian product* of S. In particular, its objects will be called *tuples over* S and denoted as $(u_0, \ldots, u_{n-1})$. Moreover, a *sequence over* S is an arrow $\mathbb{N} \to S$ and is denoted as $(u_i)_{i\in\mathbb{N}}$. Finally, a *word over* S is an object of the Kleene closure $S^\star$ which be can be defined as follows

$$S^\star \quad = \quad \bigcup_{i\in\mathbb{N}} S^i$$

We will adopt the small assumption that tuples, sequences and words can be mixed up to isomophisms and will suggest the following common notations.

The *k-component projection* of a sequence u is written u_k or $u[k]$. The *prefix of length k* of the word u is $u_{<k}$ or $(u_i)_{i<k}$. The *suffix starting at the k-th component* of u is $u_{\geq k}$ or $(u_i)_{i\geq k}$.

Furthermore, a *subsequence* (or *subword*) of u is a sequence v such that there exists a non-decreasing arrow $\phi : N \to N$ satisfying $v_k = u_{\phi(k)}$ (where $N \in \mathcal{P}(\mathbb{N})$). We insist on the fact that this is a general case from a *factor* y of u when there exists x, z such that $u = x \cdot y \cdot z$.

Let R be a binary relation and S a set. We denote $R_{|S}$ as the restriction of R over domain S, *i.e.* $R_{|S} = \{\langle x, y\rangle : x \in S \text{ and } x\ R\ y\}$. When x is an object of S, then we abbreviate R_x as $R_{\{x\}}$. In the end, we denote the following, R^0 as the reflexive closure, R^+ as the transitive closure and R^* as the reflexive-transitive closure.

1.2 Domains, recursion and fixpoints

In recursion theory, let S, S' be two sets. The set of *partial recursive functions* is $S \rightharpoonup S'$, yet the set of *total functions* is $S \to S'$. Let $f : S \rightharpoonup S'$ be a partial recursive function. We shall note $f(x) \nearrow$ if the result of the calculation is undefined and $f(x) \searrow$ if it is defined. Furthermore, the Kleene's μ-operator (also called *minimization operator*) is defined as

$$\mu x.\ \Big[P(x)\Big] \quad = \quad \mathsf{inf}\ \big\{x \in \mathbb{N} : P(x)\big\} \quad \text{if} \quad \exists x \quad P(x)$$

Finally, we will need some fixpoint theory. Let $(X, \sqsubseteq, \sqcup, \sqcap, \bot, \top)$ be a complete lattice. Recall the following notations

$$\begin{array}{rcl} \mathsf{fp}(f) & = & \big\{x \in X : f(x) = x\big\} \\ \mathsf{lfp}_x\ f & = & \mathsf{min}_{\sqsubseteq_X}\ \big\{y \in \mathsf{fp}(f) : x \sqsubseteq_X y\big\} \text{ if exists} \\ \mathsf{lfp}\ f & = & \mathsf{lfp}_\bot\ f \end{array}$$

The usage of least fixpoint operators is restricted to specific algebraic structures called *lattices*. The following is a very fundamental fixpoint theorem which gives the existence and ability to *construct* the fixpoint of a function by *successive iterations* of the function

Theorem 1. [Tarski (1955), Kleene (1938)] *If $f : X \to X$ is continuous in a complete partial order X and $\bot$ is the least element of X then* $\mathsf{lfp}\ f$ *exists and can be computed as*

$$\mathsf{lfp}\ f \quad = \quad \mathsf{sup}\ \big\{f^n(\bot) : n \in \mathbb{N}\big\}$$

2 Defining time in TESL

At most, we will stick to the following letters and fonts to give some intuitions for the reader.

clocks	$c, c_1, c_2, \ldots$
tags	τ, τ_k^c
tick indexes	k
ticks	$\langle c, k\rangle$
instant ranks	r
instants	$I_0, I_1, \ldots, I_r$
runs	(I)
ticking predicate (local)	$\uparrow_k^c$
ticking predicate (global)	$\Uparrow_r^c$
time structures	$\mathcal{I}$
TESL specifications	Φ

We give some hints about the entities that will be defined further.

clock A clock $c \in \mathcal{C}$ is an event (*i.e.* a set of possible occurrences of event). Its *time domain* — denoted as $\mathsf{dom}(c) \in \{\mathbb{U}, \mathbb{Z}, \mathbb{D}, \mathbb{Q}, \mathbb{F}\}$ — stands for a set of tags that event occurrences can be stamped with ;

tick The occurrence of an event of clock c is a tick at the k-th index described a 2-tuple $\langle c, k\rangle \in \mathcal{C} \times \mathbb{N}$. Such clock c is said to be *ticking at* local index k if predicate $\uparrow_k^c$ holds ;

tag The tag of a tick on clock c at index k is given by the partial function[1] $(\mathsf{tag}^c)_c : \mathcal{C} \to \mathbb{N} \rightharpoonup \mathsf{dom}(c)$ and abbreviated (wlog in the case of time structures at runtime) as τ_k^c;

instant An instant $I_r \in \mathcal{P}_f(\mathcal{C} \times \mathbb{N})$ (fixed rank r) is a set of clock ticks ;

run A run $(I_r)_{r\in\mathbb{N}} : \mathbb{N} \to \mathcal{P}_f(\mathcal{C} \times \mathbb{N})$ is a discrete sequence of countable and possibly infinite instants ;

[1] This partial function can also be considered to have $\mathcal{C} \times \mathbb{N}$ as domain and $\mathsf{dom}(c)$ as codomain by decurryfication. In particular, the domain in question can be restricted to an inductive set ticks_Φ as given in Appendix.

Notice that the occurence of a tick on clock c at index k is purely equivalent to the fact that there exists a tag that stamps on clock c at index k. That is to say,

$$\uparrow_k^c \ \text{holds} \quad \text{if and only if} \quad \tau_k^c \searrow$$

We can notice that ticks are stamped with two entities : the clock in which it is ticking and the local index on the latter. On the other hand, instants are stamped with global rank which indicates how time can progress.

We finally define two structures upon which we will construct a correctness assessment property in the next sections

1. A *time structure* $\mathcal{I}$ is a tuple

$$\langle (\mathsf{tag}_{\mathcal{I}}^c)_{c\in\mathcal{C}}, (I_r)_{r\in\mathbb{N}} \rangle$$

where the arrow $(\mathsf{tag}_{\mathcal{I}}^c)_{c\in\mathcal{C}} : \mathcal{C} \to \mathbb{N} \rightharpoonup \mathsf{dom}(c)$ maps a tick (*i.e.* a clock-stamped local index) to its tag and (I) is a run.

2. A TESL *specification* Φ is a tuple

$$\langle \mathcal{C}_\Phi, \mathsf{dom}_\Phi, (\mathsf{tag}_\Phi^c)_{c\in\mathcal{C}}, \mathcal{R}^t, \mathcal{R}^c \rangle$$

where $\mathcal{C}_\Phi$ is a set of clocks, $\mathsf{dom}_\Phi(c)$ is the time domain of clock c, the arrow $(\mathsf{tag}_\Phi^c)_{c\in\mathcal{C}} : \mathcal{C} \to \mathbb{N} \rightharpoonup \mathsf{dom}(c)$ maps a tick to its tag, $\mathcal{R}^t$ is the set of tag relations and $\mathcal{R}^c$ the set of causality relations.

3 Correctness properties of runs

Let Φ be a TESL specification. The correctness property of a TESL run depends on a TESL specification. For instance, it would be possible to rephrase the correctness assessment problem into a model-checking problem by considering that a time structure $\mathcal{I}$ has to satisfy a specification Φ

$$\mathcal{I} \quad \models \quad \Phi$$

As a matter of fact, the TESL solver answers a more general problem which is the TESL-satisfiability problem by returning a run that satisfies the specification (*i.e.* a model such that the logic holds) but under the strict condition that each (generated or not) tick has a *deterministic* computable tag.

We are interested in giving a correctness assessment predicate for any time structure. Such property can be given by the conjunction of the properties shown in the next subsections.

Assume $\mathcal{I} = \langle \dots \rangle$ is a time structure and $\langle \dots \rangle$ a TESL specification. We introduce a small and convenient notation that means that clock c is ticking at global runtime rank r (whether specifying the tick local index or not)

$$\begin{aligned} \Uparrow_r^{\langle c,k\rangle} \quad &\text{if and only if} \quad (c,k) \in I_r \\ \Uparrow_r^{c} \quad &\text{if and only if} \quad \exists k \quad (c,k) \in I_r \end{aligned}$$

3.1 Clocks, ticks and run consistency

As seen right above, the tag arrow of a clock is defined seperately whether specification or runtime. Indeed, the specification gives a constraint about a minimal quantity of ticks. Thus, a time structure has to contain all of ticks that are specified but perhaps even more when additional ticks are generated by the solver.

We first check that clocks and their domains are preserved

$$\mathcal{C}_\mathcal{I} = \mathcal{C}_\Phi \tag{1}$$

$$\mathsf{dom}_\mathcal{I} = \mathsf{dom}_\Phi \tag{2}$$

Then, we check that ticks are preserved at runtime when only specified as sporadic or periodic,

$$\forall c \in \mathcal{C} \quad \forall k \quad \mathsf{tag}_\Phi^c(k) \searrow \quad \text{implies} \quad \mathsf{tag}_\mathcal{I}^c(k) \searrow \tag{3}$$

It is not enough to check that runtime contains enough ticks as specified. We also need to check that tags are correctly and increasingly scattered. Indeed, tags that were previously stamped to ticks at specification may be shifted or even scattered at runtime when additional ticks are generated or stuttered (see example in Section 4.2). Hence,

$$\forall c \in \mathcal{C} \quad \exists N = \Big[0 \ ; \ |\mathsf{dom}\left(\mathsf{tag}_\Phi^c\right)|\Big] \quad \exists S : N \to \mathbb{N} \ \text{non-decreasing} \quad \mathsf{tag}_\mathcal{I}^c\big(S(k)\big) = \mathsf{tag}_\Phi^c(k) \tag{4}$$

Remark 1. *In such case, if there exists $N \subseteq \mathbb{N}$ such that $S = \mathsf{Id}_{|N}$ then no additional ticks have been generated between specification and runtime.*

Then, we need to state that ticks at runtime are necessarily and sufficiently covered by/distributed over instant sequences. That is, instants stand for a disjoint partitions of the set of ticks,

$$\forall c \quad \forall \langle c,k\rangle \in \mathsf{ticks}_{\mathcal{I}}(c) \quad \exists! r \qquad \Uparrow_r^{\langle c,k\rangle} \tag{5}$$

Moreover, we check that a clock ticking in a instant has only one very unique tick occurence

$$\forall r \quad \forall \langle c,k\rangle, \langle c,k'\rangle \quad \left[\Uparrow_r^{\langle c,k\rangle} \quad \mathsf{and} \quad \Uparrow_r^{\langle c,k'\rangle} \right] \quad \mathsf{implies} \quad k = k' \tag{6}$$

We also check that precedence relations between local indexes and global ranks are preserved in both ways. That is

$$\forall r,s \quad \forall \langle c,k\rangle, \langle c,k'\rangle \quad \left[\Uparrow_r^{\langle c,k\rangle} \quad \mathsf{and} \quad \Uparrow_s^{\langle c,k'\rangle} \quad \mathsf{and} \quad r \leq s \right] \quad \mathsf{if\ and\ only\ if} \quad k \leq k' \tag{7}$$

Lemma 1. *Progressing in global rank leaves no hole during progress of local indexes*

$$\forall r \quad \forall \langle c,k\rangle \quad \Uparrow_r^{\langle c,k\rangle} \quad \mathsf{implies} \quad \mu k'_{k'>k}.\ \left[\exists r' > r. \quad \Uparrow_{r'}^{\langle c,k'\rangle} \right] \quad = \quad k+1 \tag{8}$$

Proof. Draft. Instants partitioning property. Order preservation by distribution over instants. □

Lemma 2. *Let $c \in \mathcal{C}$ be a clock. The arrow $\mathsf{tag}^c : \mathbb{N} \to \mathsf{dom}(c)$ is monotonic.*

3.2 Implications

Consider the following clock implication

$$c_1 \ \mathsf{implies}\ c_2$$

The correctness predicate can be stated as

$$\forall r \quad \Uparrow_r^{c_1} \quad \Rightarrow \quad \Uparrow_r^{c_2} \tag{9}$$

We need to check that when c_1 ticks then c_2 also has to tick at the very same instant I_r.

3.3 Tags and time scales

To proceed with tags relations which allow to synchronize ticks and give a time-triggered behavior, specification gives a set of pair (d, r) such that

$$\begin{aligned} d \circ r \circ d &= d \\ r \circ d \circ r &= r \end{aligned}$$

We define a partial matrix of tag relations where two functions d and r are given such that each

$$\begin{aligned} R_{c_1,c_2}(\tau_1,\tau_2) &= d_{c_1,c_2}(\tau_1) = \tau_2 \quad \mathsf{or} \quad r_{c_1,c_2}(\tau_2) = \tau_1 \\ \equiv_{c_1,c_2}(\tau_1,\tau_2) &= \begin{cases} \tau_1 = \tau_2 & \text{if } c_1 = c_2 \\ (R_{c_1,c_2})^+ & \text{if } d_{c_1,c_2} \text{ and } r_{c_1,c_2} \text{ exists in } \mathcal{R}^t \\ \mathsf{true} & \text{otherwise} \end{cases} \end{aligned}$$

The correctness property for tag relations is

$$\forall r \quad \forall \langle k_1,c_1\rangle, \langle k_2,c_2\rangle \in I_r \quad \left[\Uparrow_r^{\langle k_1,c_1\rangle} \quad \mathsf{and} \quad \Uparrow_r^{\langle k_2,c_2\rangle} \right] \quad \mathsf{implies} \quad \tau_{k_1}^{c_1} \quad \equiv_{c_1,c_2} \quad \tau_{k_1}^{c_1} \tag{10}$$

Tag relations give constraints on the way ticks are distributed over instants. In particular, we need to check that every tag of ticks satisfy a coincidence relation $\equiv_{c_1,c_2}$ that is either equality $=$ when same clocks, either the relation $\mathcal{R}$ (specified in Φ) or its approximate "inverse" both transitive-closed, either true when no constraint matches with clocks in question.

3.4 Filtered implication

Consider the following filtered implication

$$c_1 \text{ filtered by } s, k \ (rs, rk)^\star \text{ implies } c_2$$

We additionally define a unary predicate over instant ranks to specify the filter predicate as

$$\begin{aligned} \mathsf{filter}(x) &= \left[\mathsf{false}^s \ \mathsf{true}^k \ (\mathsf{false}^{rs} \ \mathsf{true}^{rk})^\omega\right][x] \\ &\equiv \ s \le x < s+k \quad \text{or} \quad \exists n \ge 0 \quad s+k+n(rs+rk)+rs \le x < s+k+(n+1)(rs+rk) \end{aligned}$$

Finally, we check that

$$\forall r \quad \text{s.t. } \mathsf{filter}(r) \text{ holds} \quad \Uparrow_r^{c_1} \quad \Rightarrow \quad \Uparrow_r^{c_2} \tag{11}$$

The filtered implication works the same way as the implication but considers only a part of ticks described by a pattern. The pattern in question is described by predicate filter which decides if it is necessary to filter or not at a specific rank r.

3.5 Delayed implication

Consider the following delayed implication

$$c_1 \text{ delayed by } \delta \text{ on } c_2 \text{ implies } c_3$$

We first define an extraction operator which maps a natural integer n into the rank of n-th occurence of a tick on a given clock

$$X_c^{(I)}(n) \quad = \quad \begin{cases} \mu x. \left[\Uparrow_x^c \right] & \text{if } n = 0 \\ \mu x. \left[\Uparrow_x^c \quad \text{and} \quad x > X_c^{(I)}(n-1) \right] & \text{otherwise} \end{cases}$$

The correctness property for delayed implications can be stated as

$$\forall r \quad \Uparrow_r^{c_1} \quad \forall r' > r \quad r' = X_{c_2}^{I_{>r}}\big(\delta - 1\big) + r + 1 \quad \text{and} \quad \Uparrow_{r'}^{c_3} \tag{12}$$

In particular, this property needs the μ-operator as it is necessary to count the number of delayed ticks. This is handled by function X_c^I which catches the next rank when c ticks on run I and jumps to the other one with minimization operator μ.

Concerning the variant with the immediate modality, we have

$$\forall r \quad \Uparrow_r^{c_1} \quad \forall r' > r \quad r' = X_{c_2}^{I_{\ge r}}\big(\delta - 1\big) + r \quad \text{and} \quad \Uparrow_{r'}^{c_3}$$

3.6 Sustained implication

Consider the following sustained implication

$$c_{\mathsf{master}} \text{ sustained from } c_{\mathsf{begin}} \text{ to } c_{\mathsf{end}} \text{ implies } c_{\mathsf{slave}}$$

We state the correctness property as

$$\forall r_1 \quad \Uparrow_{r_1}^{\mathsf{begin}} \quad \forall r_2 = \mu x_{r_1 < x \le +\infty}. \left[\Uparrow_x^{\mathsf{end}} \right] \quad \forall r_1 \le r \le r_2 \quad \Uparrow_r^{\mathsf{master}} \quad \Rightarrow \quad \Uparrow_r^{\mathsf{slave}} \tag{13}$$

In this case, we need to look for the instant rank when c_{end} ticks that is strictly greater that the one when c_{begin} ticks to know the interval in which the implication holds. This is done with the minimization operator. Then one just needs to check causalities inside that interval $[r_1; r_2]$.

3.7 When implication

Consider the following sustained implication

$$c_1 \text{ when } c_2 \text{ implies } c_3$$

The correctness property is

$$\forall r \quad \left[\Uparrow_r^{c_1} \quad \text{and} \quad \Uparrow_r^{c_2} \right] \quad \Rightarrow \quad \Uparrow_r^{c_3} \tag{14}$$

The when implication can be associated to a logical $\wedge$ in propositional logic. Hence, the correctness property only needs to add another assumption on clock c_2 to hold.

3.8 Await implication

Consider the following await implication

$$\mathsf{await}\ c_1\ c_2\ \ldots\ c_n\ \mathsf{implies}\ c_{\mathsf{slave}}$$

We first define an extraction operator A which maps a natural integer into the greatest mininal rank where all clocks in set C have ticked at least once and the slave clock is supposed to tick. We can notice that this extractor uses μ over each clocks to fetch the soonest tick, then the sup keeps the latest to state when the slave clock is supposed to tick. Afterall, the next object of the sequence jumps after the previous and keeps running this way

$$A^C(n) \quad = \quad \begin{cases} \mathsf{sup} \left\{ \mu x. \left[\Uparrow_x^c \right] \quad : \quad c \in C \right\} & \text{if } n = 0 \\ \mathsf{sup} \left\{ \mu x. \left[\Uparrow_x^c \quad \mathsf{and} \quad x > A^C(n-1) \right] \quad : \quad c \in C \right\} & \text{otherwise} \end{cases}$$

We can now define the correctness predicate as

$$\forall x \geq 0 \quad \exists r = A^{\{c_1,\ldots,c_n\}}(x) \quad \Uparrow_r^{\mathsf{slave}} \tag{15}$$

In this case, we need to mix the μ-operator with the supremum as we need to construct minimal successive partitions where all clocks in C ticked at least once. Then A^C catches this partition and returns the highest rank where last clock has ticked to allow the release of the await deadlock.

3.9 Time delayed implication

Consider the following time delayed implication

$$c_{\mathsf{master}}\ \mathsf{time\ delayed\ by}\ \delta_t\ \mathsf{on}\ c_{\mathsf{measuring}}\ \mathsf{implies}\ c_{\mathsf{slave}}$$

In this case and contrary to above, we will need to keep the clock local index as we will be reasoning on tags. Correctness can be expressed as

$$\forall r \quad \exists k \quad \Uparrow_r^{\langle \mathsf{master},k \rangle} \quad \exists r' > r \quad \exists k' \quad \Uparrow_{r'}^{\langle \mathsf{slave},k' \rangle} \quad \mathsf{such\ that} \quad \tau_{k'}^{\mathsf{slave}} = \tau_k^{\mathsf{master}} + \delta_t \tag{16}$$

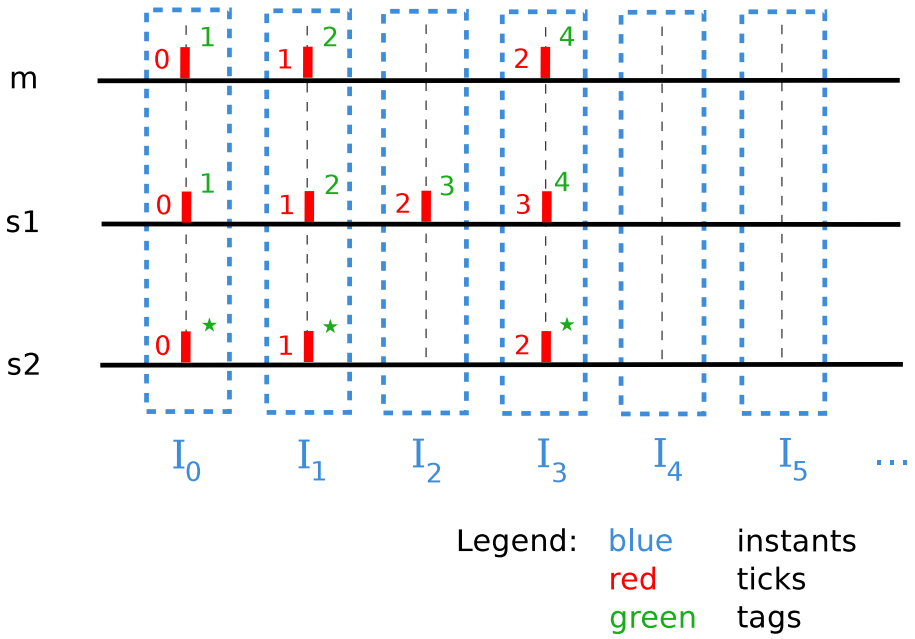
4 Example

4.1 Example 1 : a tag relation and a Unit clock

```
int-clock m sporadic  1, 2, 4
int-clock s1 sporadic 1, 2, 3, 4
unit-clock s2

tag relation m = s1
m implies s1
m implies s2
```

There exists a run that satisfies the specification above returned by the TESL solver and given by



That is to say, this run can be described by the following equations.
The set of clocks is

$$\mathcal{C} = \{m, s_1, s_2\}$$

The sets of ticks at specification and runtime are different as stated below

$$\begin{aligned} \mathsf{ticks}_\Phi &= \{m\} \times \{0;1;2\} \cup \{s_1\} \times \{0;1;2;3\} \\ \mathsf{ticks}_\mathcal{I} &= \{m\} \times \{0;1;2\} \cup \{s_1\} \times \{0;1;2;3\} \cup \{s_2\} \times \{0;1;2\} \end{aligned}$$

Ticks still keep the same tags as deployed in specification (no shift). New tags are added upon them and stamp ticks as

$$\begin{aligned} \mathsf{tag}^m_\mathcal{I}(0) = \tau^m_0 &= 1 \\ \tau^m_1 &= 2 \\ \tau^m_2 &= 4 \\ &\dots \\ \tau^{s_2}_0 &= \star \\ &\dots \end{aligned}$$

Finally, the run can be described as the sequence (I_n)

$$\begin{aligned} I_0 &= \{\langle m, 0\rangle, \langle s_1, 0\rangle, \langle s_2, 0\rangle\} \\ I_1 &= \{\langle m, 1\rangle, \langle s_1, 1\rangle, \langle s_2, 1\rangle\} \\ I_2 &= \{\langle s, 2\rangle\} \\ I_3 &= \{\langle m, 2\rangle, \langle s, 3\rangle, \langle s, 2\rangle\} \\ I_{n\geq 4} &= \varnothing \end{aligned}$$

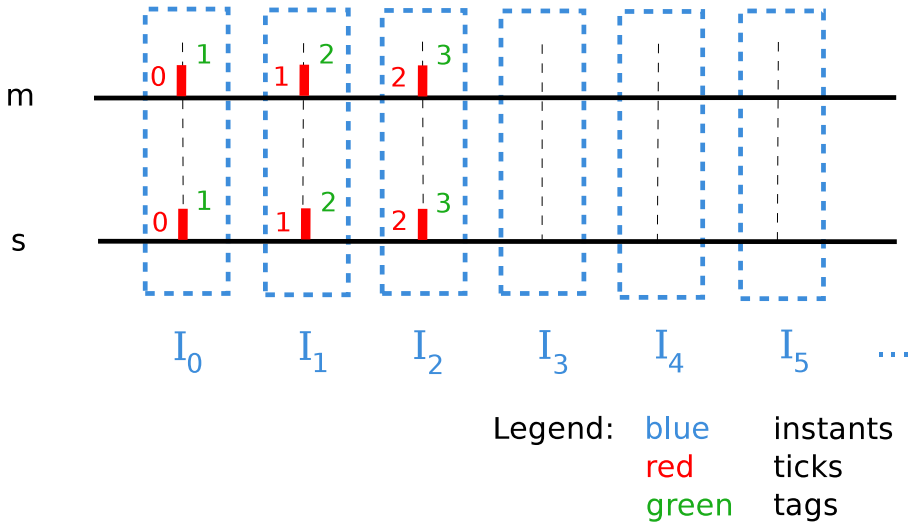
4.2 Example 2 : created ticks and shifted tags

```
int-clock m sporadic 1, 2, 3
int-clock s sporadic 2

tag relation m = s
m implies s
```

Notice that at specification time, clock s had only one tick called $\langle s, 0\rangle$ and its tag was $\mathsf{tag}^s_\phi(0) = 2$. In this case, the solver has to satisfy the identity tag relation between m and s and thus is constrained to create two ticks $\langle s, 1\rangle$ and $\langle s, 2\rangle$. It appears that tags that were previously assigned to ticks are now shifted and a new tag deployment is generated by the solver.

In particular, the solver returns the following time structure



Hence, the new tag deployment for clock s is (tags for m remain unchanged)

$$\begin{aligned} \mathsf{tag}^s_{\mathcal{I}}(0) &= 1 \\ \mathsf{tag}^s_{\mathcal{I}}(1) &= 2 \\ \mathsf{tag}^s_{\mathcal{I}}(2) &= 3 \end{aligned}$$

Now, tick $\langle s, 1\rangle$ has tag 2, instead of $\langle s, 0\rangle$ as seen above at specification time. This behavior is checked and validated by equation (3).

5 Appendix 1 : Frontend for the TESL syntactic part

Let $\square \in \{\mathbb{U}, \mathbb{Z}, \mathbb{D}, \mathbb{Q}, \mathbb{F}\}$ be a set symbol. We denote operations $+_\square$ and $\times_\square$ that allow time domain $\langle \square, +_\square, \times_\square \rangle$ to be a ring. The set of ticks ticks_Φ defined at specification is inductively defined as follows,

1. if $\square$-clock $c \in \Phi$,

$$\mathsf{ticks}_{\Phi|c} = \varnothing$$

2. if $\square$-clock c sporadic $\tau_1, \ldots, \tau_n \in \Phi$,

$$\begin{aligned} \mathsf{ticks}_{\Phi|c} &= \Big\{\langle c, 0\rangle, \langle c, 1\rangle, \ldots, \langle c, n\rangle\Big\} \\ \forall 1 \leq k \leq n \quad \mathsf{tag}^c_\Phi(k) &= \tau_k \end{aligned}$$

3. if $\square$-clock c periodic a offset $b \in \Phi$,

$$\begin{aligned} \mathsf{ticks}_{\Phi|c} &= \Big\{\langle c, k\rangle \quad : \quad k \in \mathbb{N}\Big\} \\ \forall k \in \mathbb{N} \quad \mathsf{tag}^c_\Phi(k) &= a \times_\square k +_\square b \end{aligned}$$

6 Appendix 2 : Embedding approximation in TESL tag relations

Checking tag relations need to consider and understand ring structures. In particular, for any tag relation of the kind tag relation $c_1 = a \times_\square c_2 +_\square b$, the relation constrains ticks of the very same instant to satisfy the following constraint or its "inverse"

$$\begin{aligned} d \ : \ \mathsf{dom}(c_1) &\rightarrow \mathsf{dom}(c_2) \\ x &\mapsto a \times_\square x +_\square b \end{aligned}$$