Discounting in Time

Uli Fahrenberg and Kim G. Larsen

Dept. of Computer Science
Aalborg University
Denmark
Email: {uli,kgl}@cs.aau.dk

Abstract

This paper deals with the issue of discounting in weighted timed transition systems. Discounting provides a way to model optimal-cost problems for infinite runs and has applications in optimal scheduling and other areas.

We show that when postulating a certain natural additivity property for the discounted weights of runs, there is essentially only one possible way to introduce a discounting semantics. Our proof relies on the fact that a certain functional equation essentially only has one solution, for which we provide an elementary proof.

Key words: Timed transition systems, timed automata, weighted timed automata, priced timed automata, discounting

1 Introduction

The notion of *timed transition system* is fundamental to the modeling and analysis of real-time systems, where it serves as a basic model and is usually instantiated using some finitary formalism such as timed automata [2], time Petri nets [16], timed CCS [17] or another. Timed automata in particular have successfully been used in dealing with scheduling problems [1,4,8,9,14].

To address the issue of *optimal* scheduling, the notions of *weighted*, or *priced*, timed automata, and of *weighted timed transition systems* as underlying semantic model, have been introduced [5,3]. Recently, interest has risen in applying these models to issues of *infinite* optimal scheduling, where the aim is to devise infinite schedules for a given system which are optimal in some sense, see *e.g.* [6,7,12] for some examples.

When dealing with infinite optimal scheduling, one has to take care that the weight accumulated during an infinite execution is *finite*, at least in the interesting cases. This can be accomplished through the use of quotients as in [6], *e.g.* quotients of accumulated weight with elapsed time, or through the notion of *discounting*. Under this principle, the contribution of a certain

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

part of the behaviour to the overall weight depends on how far into the future this part takes place — events further in the future are discounted in their contribution to the accumulated weight. Discounting is a well-known principle in economics, and has been used in the context of timed automata *e.g.* in [13].

Several choices are available when discounting; one can discount by steps as *e.g.* in [15,18] or by elapsed time as *e.g.* in [12]. Under the former version, events are discounted by the number of steps it takes to reach them; under the latter, by the time elapsed until their occurrence.

It is the purpose of this note to show that if one requires a certain natural additivity property for the discounted accumulated weights of runs, then there is essentially only one way to introduce discounting into the weighted timed transition systems formalism.

The first author wishes to express his gratuity to Arne Jensen and Martin Raussen from the mathematical sciences department at Aalborg University for interesting discussions on the subject of Section 3.

2 Discounting in weighted timed transition systems

Definition 2.1 A *weighted timed transition system* (S, T_s, T_d, w, r) consists of a set of states S, a set of switch transitions $T_s \subseteq S \times S$, a set of delay transitions $T_d \subseteq S \times \mathbb{R}_{\geq 0} \times S$, transition weights $w : T_s \to \mathbb{R}$, and state weight rates $r : S \to \mathbb{R}$. The delay transitions are subject to the following axioms:

- $(s, 0, s) \in T_d$ for all $s \in S$,
- for any $(s_1, x, s_2) \in T_d$ and $(s_2, y, s_3) \in T_d$, also $(s_1, x + y, s_3) \in T_d$,
- for any $(s_1, x, s_2) \in T_d$ and $y \leq x$, there exists $s_3 \in S$ such that $(s_1, y, s_3) \in T_d$ and $(s_3, x - y, s_2) \in T_d$, and
- whenever $(s, x, s_1) \in T_d$ and $(s, x, s_2) \in T_d$, then $s_1 = s_2$.

Transitions are usually written $s \to s'$ and $s \xrightarrow{x} s'$ instead of $(s, s') \in T_s$ respectively $(s, x, s') \in T_d$; a general (switch or delay) transition will be denoted $\xrightarrow{\square}$. The weight $w(e)$ of a switch transition $e = s \to s'$ can be used to model an amount of resources required for, or (if negative) of resources gained by, taking this transition, and the weight rate $r(s)$ of a state s to measure the amount of resources per time unit required to stay (or again, gained by staying) in that state.

We say that a weighted timed transition system is *delay-enabled* if there is $s \in S$ and $x \in \mathbb{R}_{>0}$ such that $(s, x, s') \in T_d$ for some (necessarily unique) $s' \in S$. This is a natural property; weighted timed transition systems which are not delay-enabled are discrete.

Weighted timed transition systems arise naturally as the semantics of weighted timed automata, see [6,12]. The precise formulation of this shall be of little concern for us here; we only note that the weighted timed transition systems arising this way obey the property of having *delay-invariant* *weight rates*:

If (S, T_s, T_d, w, r) arises as the semantics of a weighted timed automaton A, then the states in S are pairs consisting of locations of A and valuations of its clocks. The weight rates depend on the locations only, and a delay only changes the valuation component of a state, hence if $(s, x, s') \in T_d$, then $r(s) = r(s')$.

A *path* in a weighted timed transition system $\mathcal{T}$ is a (finite or infinite) sequence $s_0 \xrightarrow{\square} s_1 \xrightarrow{\square} s_2 \xrightarrow{\square} \cdots$ of switch and delay transitions in $\mathcal{T}$. The set of all finite paths in $\mathcal{T}$ is denoted $\mathcal{P}_\mathcal{T}$, and if $\pi_1 = s_0^1 \xrightarrow{\square} s_1^1 \xrightarrow{\square} \cdots \xrightarrow{\square} s_n^1, \pi_2 = s_0^2 \xrightarrow{\square} s_1^2 \xrightarrow{\square} \cdots \xrightarrow{\square} s_m^2 \in \mathcal{P}_\mathcal{T}$ with $s_n^1 = s_0^2$, then $\pi_1 \circ \pi_2 = s_0^1 \xrightarrow{\square} s_1^1 \xrightarrow{\square} \cdots \xrightarrow{\square} s_n^1 \xrightarrow{\square} s_1^2 \xrightarrow{\square} \cdots \xrightarrow{\square} s_m^2$ denotes their concatenation.

When employing weighted timed transition systems for optimal scheduling, one uses the weight and rate functions to introduce an *accumulated weight function* $W : P_\mathcal{T} \to \mathbb{R}$. When dealing with problems in *reachability* optimal scheduling as *e.g.* in [3,5], this suffices, but for *infinite* optimal scheduling one has to introduce accumulated weights of infinite paths as a limit of the accumulated weights of its finite prefixes, *cf.* [6,12,15].

There are different ways to ensure that the above-mentioned limit exists for most paths, notably the quotient approach of [6] and the (different) discounting approaches of [12,15] and others. The technical definitions are of little importance here, as we shall be interested in a specific property of the accumulated weight function rather than in its precise formulation. (Compare this to [15], where the authors also show some properties for a whole class of accumulated weight functions rather than for concrete examples only.)

Definition 2.2 An accumulated weight function $W : \mathcal{P}_\mathcal{T} \to \mathbb{R}$ is said to be *discounted in time* if there exists a function $g : \mathbb{R}_{\geq 0} \to \mathbb{R}$ such that

$$W(e \circ \pi) = \begin{cases} W(e) + W(\pi) & \text{if } e \in T_s \\ W(e) + g(x) W(\pi) & \text{if } e = s \xrightarrow{x} s' \in T_d \end{cases}$$

for all transitions e and compatible paths π.

Hence an accumulated weight function which is discounted in time has the property that weights after switches are not discounted; no time passes during a switch, and after a delay of x time units, weights are discounted by a value $g(x)$ according to a (usually decreasing) *discount function* g.

The above definition imposes no restrictions on the weights $W(e)$ for switches $e \in T_s$, but it is natural to let $W(e)$ depend on the weight $w(e)$. As we shall soon see, the situation is quite different for accumulated weights $W(e)$ for $e \in T_d$; the definition imposes severe restrictions on how these can be defined. Note that the definition also implies that an accumulated weight function which is discounted in time enjoys a useful recursive property which can be employed in computations.

We say that an accumulated weight function $W : \mathcal{P}_\mathcal{T} \to \mathbb{R}$ which is discounted in time, in a weighted timed transition system $\mathcal{T}$ with delay-invariant weight rates, is *natural* if $W(s \xrightarrow{x} s')$ only depends on s and x for all delay transitions $s \xrightarrow{x} s'$, and if both W and its associated discount function g are non-trivial, *i.e.* not identical to zero.

Theorem 2.3 *Let $\mathcal{T}$ be a delay-enabled weighted timed transition system with delay-invariant weight rates and $W : \mathcal{P}_\mathcal{T} \to \mathbb{R}$ a natural accumulated weight function which is discounted in time, with associated discount function $g : \mathbb{R}_{\geq 0} \to \mathbb{R}$. Then there exist $\lambda \in \mathbb{R}_{>0}$ and $\alpha : S \to \mathbb{R}$ such that $W(s \xrightarrow{x} s') = \alpha(s) \int_0^x \lambda^t dt$ and $g(x) = \lambda^x$ for all $s \in S$ and all delay transitions $s \xrightarrow{x} s'$.*

The above theorem implies that if one is interested in a recursive property as of Definition 2.2, then there is essentially only *one* possible definition of accumulated weight of delays. Incidentally, this is also the definition given in [12]. The proof of the theorem relies on the solution of a certain functional equation which is given below.

3 Proof

Theorem 2.3 will be an easy corollary of the below theorem on a certain functional equation. The equation is a generalization of the well-known Cauchy equation $f(x+y) = f(x) + f(y)$ for which a full solution was given in Cauchy's [11], and the theorem shows that also its solution is a generalization of Cauchy's. Moreover, the proof given uses only methods already available to Cauchy in 1821; the authors would be interested to see whether the equation can be solved more easily using some more modern methods.

Theorem 3.1 *Continuous functions $f, g : \mathbb{R}_{\geq 0} \to \mathbb{R}$ satisfy the functional equation*

$$f(x+y) = f(x) + g(x) f(y) \tag{1}$$

if and only if one of the following properties holds:

- $f(x) \equiv 0$,
- $f(x) \equiv k \neq 0$ *and* $g(x) \equiv 0$, *or*
- $f(x) = \alpha \int_0^x \lambda^t dt$ *and* $g(x) = \lambda^x$ *for some* $\alpha, \lambda \in \mathbb{R}$ *with* $\alpha \neq 0$ *and* $\lambda > 0$.

Proof.

§1. The function $f(x) \equiv 0$ is a solution of equation (1) for all functions g. If $f(x) \equiv k \neq 0$, then $k = k + g(x)k$ and hence $g(x) = 0$ for all x. On the other hand, if $g(x_0) = 0$ for some x_0, then $f(x_0 + y) = f(x_0)$ for all y, hence f is constant. This gives the trivial solutions in the first two items of the theorem.

§2. The special case $g(x) \equiv 1$ is treated in [11]; for this, the functions $f(x) = \alpha x$, for $\alpha \in \mathbb{R}$, are the only solutions. We can assume from now on that $f(x) \neq 0$ for some x, $g(x) \neq 1$ for some x, and that $g(x) \neq 0$ for all x.

§3. Let x_0 be such that $f(x_0) = 0$. Then $f(x_0 + y) = g(x_0)f(y)$ and $f(y + x_0) = f(y)$ for all y, hence $g(x_0) = 1$ by the assumptions in §2.

On the other hand, assume x_0 is such that $g(x_0) = 1$. Then $f(x_0+y) = f(x_0)+f(y)$ and $f(y+x_0) = f(y)+g(y)f(x_0)$ for all y, whence $f(x_0) = 0$ by our assumptions.

§4. By substituting $y = 0$ in equation (1), we get $f(x) = f(x) + g(x)f(0)$ for all x, hence $f(0) = 0$ by the assumptions in §2. By §3, $g(0) = 1$. With our assumptions, this in turn implies that $g(x) > 0$ for all x — for if $g(x) < 0$ for some x, then by continuity also $g(y) = 0$ for some y, which we have precluded in §2.

§5. Let $x > 0$, then by induction on the equation $f((k + 1)x) = f(kx) + g(kx)f(x)$ we get

$$f(kx) = f(x)\Big(\sum_{i=0}^{k-1} g(ix)\Big) \tag{2}$$

for all $k \in \mathbb{N}_+$. On the other hand, a similar induction on the equation $f((k + 1)x) = f(x) + g(x)f(kx)$ leads to

$$f(kx) = f(x)\Big(\sum_{i=0}^{k-1} (g(x))^i\Big) \tag{3}$$

for all $k \in \mathbb{N}_+$.

§6. Let $x_0 > 0$ and assume that $f(x_0) = 0$. Then by equation (3), $f(kx_0) = 0$ for all $k \in \mathbb{N}_+$.

Now let $n, k \in \mathbb{N}_+$ and $x_1 = \frac{n}{k}x_0$, then

$$0 = f(kx_1) = f(x_1)\Big(\sum_{i=0}^{k-1} (g(x_1))^i\Big) \tag{4}$$

by equation (3). Assume $f(x_1) \neq 0$, then $g(x_1) \neq 1$ by §3. Hence we can multiply equation (4) by $1 - g(x_1)$, to arrive at $1 - ((g(x_1))^k = 0$. As $g(x_1) > 0$ and $g(x_1) \neq 1$, this is impossible.

We have seen that $f(x_0) = 0$ implies $f(\frac{n}{k}x_0) = 0$ for all $n, k \in \mathbb{N}_+$, hence $f(\alpha x_0) = 0$ for all $\alpha \in \mathbb{R}_{>0}$ by continuity. But then $f(x) = 0$ for all $x > 0$, which we have precluded in §2. We must thus have $f(x) \neq 0$ for all $x \neq 0$; by §3, also $g(x) \neq 1$ for all $x \neq 0$.

§7. For $x \neq 0$ and $k \in \mathbb{N}_+$, we can combine equation (2) and equation (3) and divide by $f(x)$ to get

$$\sum_{i=0}^{k-1} g(ix) = \sum_{i=0}^{k-1} (g(x))^i$$

By induction this implies that for all $k \in \mathbb{N}_+$,

$$g(kx) = (g(x))^k \tag{5}$$

§8. Let $\lambda = g(1)$. Setting $x = 1$ in equation (5), we see that $g(k) = \lambda^k$ for all $k \in \mathbb{N}_+$. If $x = \frac{n}{k}$ for $n, k \in \mathbb{N}_+$ in the same equation, then $(g(x))^k = g(kx) = \lambda^n$, hence $g\big(\frac{n}{k}\big) = \lambda^{\frac{n}{k}}$ for all $n, k \in \mathbb{N}_+$. By continuity, $g(x) = \lambda^x$ for all $x \in \mathbb{R}_{>0}$.

§9. Let $\beta = f(1)$. Setting $x = 1$ in equation (3) gives

$$f(k) = \beta\Big(\sum_{i=0}^{k-1} \lambda^i\Big) = \beta\frac{1-\lambda^k}{1-\lambda}$$

for all $k \in \mathbb{N}_+$. If $x = \frac{n}{k}$ for $n, k \in \mathbb{N}_+$ in the same equation, then

$$\beta\frac{1-\lambda^n}{1-\lambda} = f(kx) = f(x)\Big(\sum_{i=0}^{k-1}(g(x))^i\Big) = f(x)\frac{1-\lambda^n}{1-\lambda^x}$$

and thus

$$f\big(\tfrac{n}{k}\big) = \beta\frac{1-\lambda^{\frac{n}{k}}}{1-\lambda}$$

for all $n, k \in \mathbb{N}_+$. By continuity, $f(x) = \beta\frac{1-\lambda^x}{1-\lambda}$ for all $x \in \mathbb{R}_{>0}$, and setting $\alpha = \frac{\beta \ln \lambda}{\lambda - 1}$ gives the desired result.

□

Theorem 2.3 now follows easily:

Proof of Theorem 2.3. Let $s \in S$ such that $s \xrightarrow{x_0} s' \in T_d$ for some $x_0 \in \mathbb{R}_{>0}$, then for all $x \leq x_0$ there is $s'' \in S$ such that $s \xrightarrow{x} s'' \in T_d$. Define $f_s : [0, x_0] \to \mathbb{R}$ by $f_s(x) = W(s \xrightarrow{x} s')$. Then $f_s(x+y) = f_s(x) + g(x) f_s(y)$ for all $x, y \in \mathbb{R}_{\geq 0}$ with $x + y \leq x_0$, and an easy adaptation of the proof of Theorem 3.1 shows that this implies that we have $\lambda \in \mathbb{R}_{>0}$ and $\alpha_s \in \mathbb{R}$ such that $g(x) = \lambda^x$ and $f_s(x) = \alpha_s \int_0^x \lambda^t dt$ for all $x \in [0, x_0]$.

The only cases left to consider are states $s \in S$ for which $s \xrightarrow{x} s'$ only for $x = 0$, but for these the statement of the theorem is empty. □

4 Conclusion and further work

We have shown that if one wants an accumulated weight function in a timed transition system to satisfy a certain natural additivity property, as given in Definition 2.2, then there is only very little choice left as to how to define it. In particular, neither the quotient approach of [6] nor the step-based discounting in [15,18] give rise to accumulated weight functions which are discounted in time.

To be discounted in time is not only a natural property to require of an accumulated weight function, it also implies a recursive characterization of accumulated weight. Hence one can employ a fixed point computation for finding accumulated weights of (finite or infinite) paths, and also for finding paths which are optimal in some sense. This in turn is expected to have implications for the availability of zone-based algorithms for computing accumulated weights in weighted timed automata.

References

[1] Abdeddaïm, Y., E. Asarin and O. Maler, *Scheduling with timed automata*, Theoretical Computer Science **354** (2006), pp. 272–300.

[2] Alur, R. and D. Dill, *A theory of timed automata*, Theoretical Computer Science **126** (1994), pp. 183–235.

[3] Alur, R., S. L. Torre and G. J. Pappas, *Optimal paths in weighted timed automata*, Theoretical Computer Science **318** (2004), pp. 297–322.

[4] Behrmann, G., E. Brinksma, M. Hendriks and A. Mader, *Production scheduling by reachability analysis – a case study*, in: *Proc. 19th Int. Parallel and Distributed Processing Symposium (IPDPS 2005)* (2005), p. 140a.

[5] Behrmann, G., A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn and F. Vaandrager, *Minimum-cost reachability for priced timed automata*, in: *Proc. 4th Int. Work. on Hybrid Systems: Computation and Control (HSCC'01)*, Lecture Notes in Computer Science **2034** (2001), pp. 147–161.

[6] Bouyer, P., E. Brinksma and K. G. Larsen, *Staying alive as cheaply as possible*, in: *Proc. 7th Int. Work. on Hybrid Systems: Computation and Control (HSCC'04)*, Lecture Notes in Computer Science **2993** (2004), pp. 203–218.

[7] Bouyer, P., U. Fahrenberg, K. G. Larsen, N. Markey and J. Srba, *Infinite runs in weighted timed automata with energy constraints*, in: Cassez and Jard [10], pp. 33–47.

[8] Bozga, M., A. Kerbaa and O. Maler, *Scheduling acyclic branching programs on parallel machines*, in: *Proc. 25th IEEE Real-Time Systems Symposium (RTSS 2004)* (2004), pp. 208–217.

[9] Brinksma, E., A. Mader and A. Fehnker, *Verification and optimization of a PLC control schedule*, International Journal on Software Tools for Technology Transfer **4** (2002), pp. 21–33.

[10] Cassez, F. and C. Jard, editors, "Proc. 6th Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS'08)," Lecture Notes in Computer Science **5215**, Springer, 2008.

[11] Cauchy, A.-L., "Cours d'analyse de l'Ecole royale polytechnique. Première partie: Analyse algébrique," Editions Jacques Gabay, Paris, 1989, (Reprint of the 1821 edition).

[12] Fahrenberg, U. and K. G. Larsen, *Discount-optimal infinite runs in priced timed automata*, in: *Proc. 10th Int. Work. on Verification of Infinite-State Systems (INFINITY'08)*, Electronic Notes in Theoretical Computer Science (2008), to appear.

[13] Henzinger, T. A., R. Majumdar and V. S. Prabhu, *Quantifying similarities between timed systems*, in: P. Pettersson and W. Yi, editors, *Proc. 3rd Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS 2005)*, Lecture Notes in Computer Science **3829** (2005), pp. 226–241.

[14] Hune, T., K. G. Larsen and P. Pettersson, *Guided synthesis of control programs using* UPPAAL, in: *Proc. IEEE ICDS Int. Work. Distributed Systems Verification and Validation* (2000), pp. E15–E22.

[15] Jurdzinski, M. and A. Trivedi, *Concavely-priced timed automata*, in: Cassez and Jard [10], pp. 48–62.

[16] Merlin, P. M. and D. J. Faber, *Recoverability of communication protocols: Implications of a theoretical study*, IEEE Trans. Comm. **COM-24** (1976), pp. 1036–1043.

[17] Moller, F. and C. M. N. Tofts, *A temporal calculus of communicating systems*, in: J. C. M. Baeten and J. W. Klop, editors, *Proc. 1st Int. Conf. on Concurrency Theory (CONCUR'90)*, Lecture Notes in Computer Science **458** (1990), pp. 401–415.

[18] Thrane, C., U. Fahrenberg and K. G. Larsen, *Quantitative simulations of weighted transition systems (extended abstract)*, in: T. Uustalu, J. Vain and J. Ernits, editors, *Proc. 20th Nordic Workshop on Programming Theory (NWPT'08)* (2008), pp. 97–99.