As Cheap as Possible: Efficient Cost-Optimal Reachability for Priced Timed Automata

Kim Larsen[12], Gerd Behrmann[1], Ed Brinksma[2], Ansgar Fehnker[4], Thomas Hune[3], Paul Pettersson[5], and Judi Romijn[4]

[1] Basic Research in Computer Science, Aalborg University
[2] Department of Computer Systems, University of Twente
[3] Basic Research in Computer Science, Aarhus University
[4] Computing Science Institute, University of Nijmegen,
[5] Department of Information Technology, Uppsala University

Abstract. In this paper we present an algorithm for efficiently computing optimal cost of reaching a goal state in the model of Linearly Priced Timed Automata (LPTA). The central contribution of this paper is a priced extension of so-called zones. This, together with a notion of facets of a zone, allows the entire machinery for symbolic reachability for timed automata in terms of zones to be lifted to cost-optimal reachability using priced zones. We report on experiments with a cost-optimizing extension of Uppaal on a number of examples.

1 Introduction

Well-known formal verification tools for real-time and hybrid systems, such as Uppaal [LPY97], Kronos [BDM+98] and HyTech [HHWT97], use symbolic techniques to deal with the infinite state spaces that are caused by the presence of continuous variables in the associated verification models. However, symbolic model checkers still share the "state space explosion problem" with their non-symbolic counterparts as the major obstacle for their application to non-trivial problems. A lot of research, therefore, is devoted to the containment of this problem.

An interesting idea for model checking of reachability properties that has received more attention recently is to "guide" the exploration of the (symbolic) state space such that "promising" sets of states are visited first. In a number of recent publications [Feh99,HLP00,BFH+,NY99,BM00] model checkers have been used to solve a number of non-trivial scheduling problems, reformulated in terms of reachability, viz. as the (im)possibility to reach a state that improves on a given optimality criterion. Such criteria distinguish scheduling algorithms from classical, full state space exploration model checking algorithms. They are used together with, for example, branch-and-bound techniques [AC91] to prune parts of the search tree that are guaranteed not to contain optimal solutions. This observation motivates research into the extension of model checking algorithms with optimality criteria. They provide a basis for the (cost-) guided exploration of state spaces, and improve the potential of model checking techniques for the resolution of scheduling problems. We believe that such extensions can be interesting for real-life applications of both model checking and scheduling.

Based on similar observations an extension of the timed automata model with a notion of *cost*, the *Linearly Priced Timed Automata* (LPTA), was already introduced in [BFH$^+$01]. This model allows for a reachability analysis in terms of accumulated cost of traces, i.e. the sum of the costs of the individual transitions in the trace. Each action transitions has an associated price p determining its cost. Likewise, each location has an associated rate r and the cost of delaying d time units is $d \cdot r$. In [BFH$^+$01], and independently in [ATP], computabitlity of minimal-cost reachability is demonstrated based on a cost-extension of the classical notion of regions.

Although ensuring computability, the region construction is known to be very inefficient. Tools like UPPAAL and Kronos use symbolic states of the form (l, Z), where l is a location of the timed automaton and Z is a *zone*, i.e. a convex set of clock valuations. The central contribution of this paper is the extension of this concept to that of *priced zones*, which are attributed with an (affine) linear function of clock valuations that defines the cost of reaching a valuation in the zone. We show that the entire machinery for symbolic reachability in terms of zones can be lifted to cost-optimal reachability for priced zones. It turns out that some of the operations on priced zones force us to split them into parts with different price attributes, giving rise to a new notion, viz. that of the *facets* of a zone.

The suitability of the LPTA model for scheduling problems was already illustrated in [BFH$^+$], using the more restricted *Uniformly Priced Timed Automata* (UPTA) model, admitting an efficient priced zone implementation via *Difference Bound Matrices* [Dil89]. The model was used to consider traces for the time-optimal scheduling of a steel plant and a number of job shop problems. The greater expressivity of LPTA also supports other measures of cost, like idle time, weighted idle time, mean completion time, earliness, number of tardy jobs, tardiness, etc. We take an aircraft landing problem [BKA00] as the application example for this paper.

The structure of the rest of this paper is as follows. In Section 2 we give an abstract account of symbolic optimal reachability in terms of *priced transition systems*, including a generic algorithm for optimal reachability. In Section 3 we introduce the model of linearly priced timed automata (LPTA) as a special case of the framework of Section 2. We also introduce here our running application example, the aircraft landing problem. Section 4 contains the definition of the central concept of priced zones. The operations that we need on priced zones and facets are provided in Section 5. The implementation of the algorithm, and the results of experimentation with our examples are reported in Section 6. Our conclusions, finally, are presented in Section 7.

2 Symbolic Optimal Reachability

Analysis of infinite state systems require symbolic techniques in order to effectively represent and manipulate sets of states simultaneously (see [FS01],[FS98], [ACJYK96,AJ94,Cer94]). For analysis of cost-optimality, additional information of *costs* associated with individual states needs to be represented. In this section, we describe a *general* framework for symbolic analysis of cost-optimal reachability on the abstract level of priced transition systems.

A *priced transition system* is a structure $\mathcal{T} = (S, s_0, \Sigma, \rightarrow)$, where S is a (infinite) set of states, $s_0 \in S$ is the initial state, Σ is a (finite) set of labels, and, $\rightarrow$ is a partial function from $S \times \Sigma \times S$ into the non-negative reals, $\mathbb{R}_{\geq 0}$, defining the possible transitions of the systems as well as their associated costs. We write $s \xrightarrow{a}_p s'$ whenever $\rightarrow (s, a, s')$ is defined and equals p. Intuitively, $s \xrightarrow{a}_p s'$ indicates that the system in state s has an a-labeled transition to the state s' with the cost of p. We denote by $s \xrightarrow{a} s'$ that $\exists p \in \mathbb{R}_{\geq 0}.\, s \xrightarrow{a}_p s'$, and, by $s \rightarrow s'$ that $\exists a \in \Sigma.\, s \xrightarrow{a} s'$. Now, an execution of $\mathcal{T}$ is a sequence $\alpha = s_0 \xrightarrow{a_1}_{p_1} s_1 \xrightarrow{a_2}_{p_2} s_2 \cdots \xrightarrow{a_n}_{p_n} s_n$. The *cost* of α, $\mathsf{cost}(\alpha)$, is the sum $\sum_{i \in \{1 \ldots n\}} p_i$. For a given state s, the *minimal cost* of reaching s, $\mathsf{mincost}(s)$, is the infimum of the costs of finite executions starting in the initial state s_0 and ending in s. Similar, the minimal cost of reaching a designated set of states $G \subseteq S$, $\mathsf{mincost}(G)$, is the infimum of the costs of finite executions ending in a state of G.

To compute minimum-cost reachability, we suggest the use of *priced symbolic states* of the form (A, π), where $A \subseteq S$ is a set of states, and $\pi : A \longrightarrow \mathbb{R}_{\geq 0}$ assigns (non-negative) costs to all states of A. The intention is that, reachability of the priced symbolic state (A, π) should ensure, that any state s of A is reachable with cost arbitrarily close to $\pi(s)$. As we are interested in minimum-cost reachability, π should preferably return as small cost values as possible. This is obtained by the following extension of the *post*-operators to priced symbolic states: for (A, π) a priced symbolic state and $a \in \Sigma$, $Post_a(A, \pi)$ is the priced symbolic state $(post_a(A), \eta)$, where $post_a(A) = \{s' \mid \exists s \in A.\, s \xrightarrow{a} s'\}$ and η is given by $\eta(s) = \inf\{\pi(s') + p \mid s' \in A \wedge s' \xrightarrow{a}_p s\}$. That is, η essentially gives the cheapest cost for reaching states of B via states in A, assuming that these may be reached with costs according to π. A symbolic execution of a priced transition system $\mathcal{T}$ is a sequence $\beta = (A_0, \pi_0), \ldots, (A_n, \pi_n)$, where for $i < n$, $(A_{i+1}, \pi_{i+1}) = Post_{a_i}(A_i, \pi_i)$ for some $a_i \in \Sigma$, and $A_0 = \{s_0\}$ and $\pi_0(s_0) = 0$. It is not difficult to see, that there is a very close connection between executions and symbolic executions: for any execution α of $\mathcal{T}$ ending in a state s, there is a symbolic execution β of $\mathcal{T}$, that ends in a priced symbolic state (A, π), such that $s \in A$ and $\pi(s) \leq \mathsf{cost}(\alpha)$. Dually, for any symbolic execution β of $\mathcal{T}$ ending in priced symbolic state (A, π), whenever $s \in A$, then $\mathsf{mincost}(s) \leq \pi(s)$. From this it follows that the symbolic semantics on priced symbolic states accurately captures minimum-cost reachability in the sense that $\mathsf{mincost}(G) = \inf\{\mathsf{mincost}(A \cap G, \pi) : (A, \pi) \text{ is reachable}\}$.

```
COST := ∞
PASSED := ∅
WAITING := {({s_0}, π_0)}
while WAITING ≠ ∅ do
    select (A, π) from WAITING
    if A ∩ G ≠ ∅ and minCost(A ∩ G, π) < COST then
        COST := minCost(A ∩ G, π)
    if for all (B, η) in PASSED: (B, η) ⋢ (A, π) then
        add (A, π) to PASSED
        add Post_a(A, π) to WAITING for all a ∈ Σ
return COST
```

Fig. 1. Abstract Algorithm for the Minimal-Cost Reachability Problem.

Let (A, π) and (B, η) be priced symbolic states. We write $(A, \pi) \sqsubseteq (B, \eta)$ if $B \subseteq A$ and $\pi(s) \leq \eta(s)$ for all $s \in B$, informally expressing, that (A, π) is "as big and cheap" as (B, η). Also, we denote by $minCost(A, \pi)$ the infimum costs in A w.r.t. π, i.e. $\inf\{\pi(s) \mid s \in A\}$. Now using the above notion of priced symbolic state and associated operations, an abstract algorithm for computing the minimum cost of reaching a designated set of goal states G is shown in Fig.1. It uses two data-structures WAITING and PASSED to store priced symbolic states waiting to be examined, and priced symbolic states already explored, respectively. In each iteration, the algorithm proceeds by selecting a priced symbolic state (A, π) from WAITING, checking that none of the previously explored states (B, η) are bigger and cheaper, i.e. $(B, \eta) \not\sqsubseteq (A, \pi)$, and adds it to PASSED and its successors to WAITING. In addition, the algorithm uses the global variable COST, which is initially set to ∞ and updated whenever a goal state is found that can be reached with lower cost than the current value of COST. The algorithm terminates when WAITING is empty, i.e. when no further priced symbolic states are left to be examined. When the algorithm of Fig. 1 terminates, the value of COST equals $\mathsf{mincost}(G)$. Furthermore, termination of the algorithm will be guaranteed provided $\sqsubseteq$ is a well-quasi ordering on priced symbolic states.

The above framework may be instantiated by providing concrete syntax for priced transition systems, together with data-structures for priced symbolic states allowing for computation of the *Post*-operations, $minCost$, as well as $\sqsubseteq$ (which should be well-quasi). In the following sections we provide such an instantiation for a priced extension of timed automata.

3 Priced Timed Automata

Linearly priced timed automata (LPTA) [BFH$^+$01,BFH$^+$,ATP] extend the model of timed automata [AD90] with *prices* on all edges and locations. In these models, the cost of taking an edge is the price associated with it, and the price of a location gives the cost-*rate* applied when delaying in that location.

Let $\mathbb{C}$ be a set of clocks. Then $\mathcal{B}(\mathbb{C})$ is the set of formulas that are conjunctions of atomic constraints of the form $x \bowtie n$ and $x - y \bowtie m$ for $x, y \in \mathbb{C}$, $\bowtie \in \{\leq$

$,=,\geq\}$,[1] n a natural number, and m an integer. Elements of $\mathcal{B}(\mathbb{C})$ are called clock constraints or zones over $\mathbb{C}$. $\mathcal{P}(\mathbb{C})$ denotes the power set of $\mathbb{C}$. Clock values are represented as functions from $\mathbb{C}$ to the non-negative reals $\mathbb{R}_{\geq 0}$, called clock valuations. We denote by $\mathbb{R}^{\mathbb{C}}$ the set of clock valuations for $\mathbb{C}$. For $u \in \mathbb{R}^{\mathbb{C}}$ and $g \in \mathcal{B}(\mathbb{C})$, we denote by $u \in g$ that u satisfies all constraints of g.

Definition 1 (Linearly Priced Timed Automata). *A linearly priced timed automaton $\mathcal{A}$ over clocks $\mathbb{C}$ is a tuple (L, l_0, E, I, P), where L is a finite set of locations, l_0 is the initial location, $E \subseteq L \times \mathcal{B}(\mathbb{C}) \times \mathcal{P}(\mathbb{C}) \times L$ is the set of edges, where an edge contains a source, a guard, a set of clocks to be reset, and a target, $I : L \to \mathcal{B}(\mathbb{C})$ assigns invariants to locations, and $P : (L \cup E) \to \mathbb{N}$ assigns prices to both locations and edges. In the case of $(l, g, r, l') \in E$, we write $l \xrightarrow{g,r} l'$.*

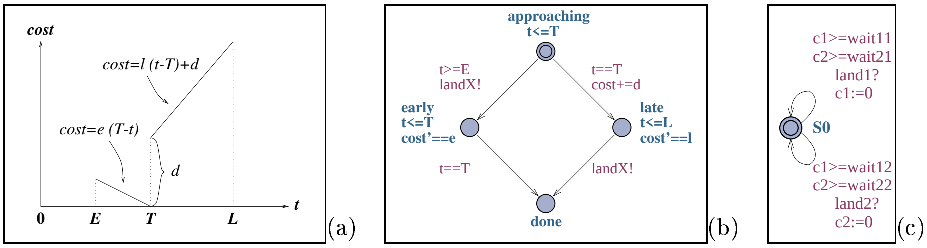

Fig. 2. Figure (a) depicts the cost of landing a plane at time t. Figure (b) shows an LPTA modelling the landing costs. Figure (c) shows an LPTA model of the runway.

The semantics of a linearly priced timed automaton $\mathcal{A} = (L, l_0, E, I, P)$ may now be given as a priced transition system with state-space $L \times \mathbb{R}^{\mathbb{C}}$ with the initial state (l_0, u_0) (where u_0 assigns zero to all clocks in $\mathbb{C}$), and with the finite label-set $\Sigma = E \cup \{\delta\}$. Thus, transitions are labelled either with the symbol δ (indicating some delay) or with an edge e (the one taken). More precisely, the priced transitions are given as follows:

- $(l, u) \xrightarrow{\delta}_p (l, u + d)$ if $\forall 0 \leq e \leq d : u + e \in I(l)$, and $p = d \cdot P(l)$,
- $(l, u) \xrightarrow{e}_p (l', u')$ if $e = (l, g, r, l') \in E$, $u \in g$, $u' = u[r \mapsto 0]$, and $p = P(e)$,

where for $d \in \mathbb{R}_{\geq 0}$, $u + d$ maps each clock x in $\mathbb{C}$ to the value $u(x) + d$, and $u[r \mapsto 0]$ denotes the clock valuation which maps each clock in r to the value 0 and agrees with u over $\mathbb{C} \setminus r$.

Example 1 (Aircraft Landing Problem). As an example of the use of LPTAs we consider the problem of scheduling aircraft landings at an airport, due to [BKA00]. For each aircraft there is a maximum speed and a most fuel efficient speed which determine an earliest and latest time the plane can land. In this

[1] For simplicity we do not deal with strict inequalities in this short version.

interval, there is a preferred landing time called target time at which the plane lands with minimal cost. The target time and the interval are shown as T and $[E, L]$ respectively in Fig. 2(a). For each time unit the actual landing time deviates from the target time, the landing cost increases with rate e for early landings and rate l for late landings. In addition there is a fixed cost d associated with late landings. In Fig. 2(b) the cost of landing an aircraft is modeled as an LPTA. The automaton starts in the initial location `approaching` and lands at the moment one of the two transitions labeled `landX!`[2] are taken. In case the plane lands too early it enters location `early` in which it delays exactly $T - t$ time units. In case the plane is late the cost is measured in location `late` (i.e. the delay in location `late` is 0 if the plane is on target time). After L time units the automaton always ends in location done. Figure 2(c) models a runway ensuring that two consecutive landings takes place with a minimum separation time. $\square$

4 Priced Zones

Typically, reachability of a (priced) timed automaton, $\mathcal{A} = (L, l_0, E, I, P)$, is decided using symbolic states represented by pairs of the form (l, Z), where l is a location and Z is a zone. Semantically, (l, Z) represents the set of all states (l, u), where $u \in Z$. Whenever Z is a zone and r a set of clocks, we denote by $Z^{\uparrow}$ and $\{r\}Z$ the set of clock valuations obtained by delaying and resetting (w.r.t. r) clock valuations from Z respectively. That is, $Z^{\uparrow} = \{u + d \,|\, u \in Z, d \in \mathbb{R}_{\geq 0}\}$ and $\{r\}Z = \{u[r \mapsto 0] \,|\, u \in Z\}$. It is well-known – using a canonical representation of zones as *Difference Bounded Matrices* (DBMs) [Dil89] – that in both cases the resulting set is again effectively representable as a zone. Using these operations together with the obvious fact, that zones are closed under conjunction, the *post*-operations may now be effectively realised using the zone-based representation of symbolic states as follows:

- $post_\delta\big((l, Z)\big) = \big(l, (Z \wedge I(l))^{\uparrow} \wedge I(l)\big)$,
- $post_e\big((l, Z)\big) = \big(l', \{r\}(Z \wedge g)\big)$ whenever $e = (l, g, r, l')$.

Now, the framework given in Section 2 for symbolic computation of minimum-cost reachability calls for an extension of our zone-based representation of symbolic states, which assigns costs to individual states. For this, we introduce the following notion of a *priced* zone, where the *offset*, Δ_Z, of a zone Z is the unique clock valuation of Z satisfying $\forall u \in Z. \forall x \in \mathbb{C}.\, \Delta_Z(x) \leq u(x)$.

Definition 2 (Priced Zone). *A priced zone $\mathcal{Z}$ is a tuple (Z, c, r), where Z is a zone, $c \in \mathbb{N}$ describes the cost of the offset, Δ_Z, of Z, and $r : \mathbb{C} \rightarrow \mathbb{Z}$ assigns a cost-rate $r(x)$ for any clock x. We write $u \in \mathcal{Z}$ whenever $u \in Z$. For any $u \in \mathcal{Z}$ the cost of u in $\mathcal{Z}$, $\mathsf{Cost}(u, \mathcal{Z})$, is defined as $c + \sum_{x \in \mathbb{C}} r(x) \cdot (u(x) - \Delta_Z(x))$.*

[2] In the example we assume that several automata $A_1, ..., A_n$ can be composed in parallel with a CCS-like parallel composition operator [Mil89] to a network $(A_1, ..., A_n) \backslash Act$, with all actions Act being restricted. We further assume that the cost of delaying in the network is the sum of the cost of delaying in the individual automata.

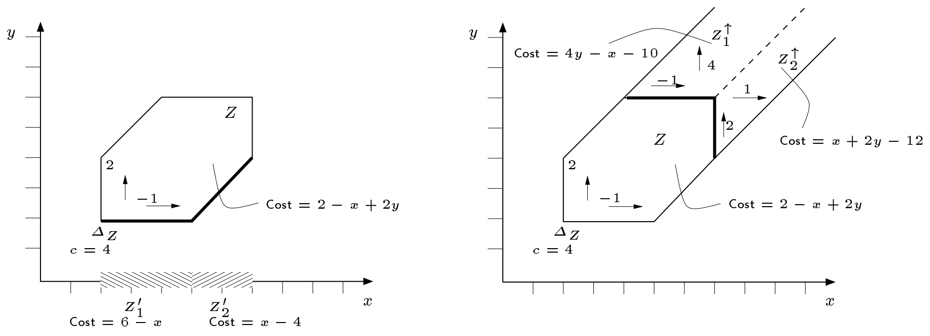

Fig. 3. A Priced Zone and Successor-Sets

Thus, the cost assignments of a priced zone define a linear plane over the underlying zone and may alternatively be described by a linear expression over the clocks. Figure 3 illustrates the priced zone $\mathcal{Z} = (Z, c, r)$ over the clocks $\{x, y\}$, where Z is given by the six constraints $2 \leq x \leq 7$, $2 \leq y \leq 6$ and $-2 \leq x - y \leq 3$, the cost of the offset ($\Delta_Z = (2, 2)$ is $c = 4$, and the cost-rates are $r(x) = -1$ and $r(y) = 2$. Hence, the cost of the clock valuation $(5.1, 2.3)$ is given by $4 + (-1) \cdot (5.1 - 2) + 2 \cdot (2.3 - 2) = 1.5$. In general the costs assigned by $\mathcal{Z}$ may be described by the linear expression $2 - x + 2y$.

Now, priced symbolic states are represented in the obvious way by pairs $(l, \mathcal{Z})$, where l is a location and $\mathcal{Z}$ a priced zone. More precisely, $(l, \mathcal{Z})$ represents the priced symbolic state (A, π), where $A = \{(l, u) \mid u \in \mathcal{Z}\}$ and $\pi(l, u) = \mathsf{Cost}(u, \mathcal{Z})$.

Unfortunately, priced symbolic states are *not* directly closed under the *Post*-operations. To see this, consider a timed automata $\mathcal{A}$ with two locations l and m and a single edge from l to m with trivial guard (*true*) and resetting the clock y.. The cost-rate of l is 3 and the transition has zero cost. Now, let $\mathcal{Z} = (Z, c, r)$ be the priced zone depicted in Fig. 3 and consider the associated priced symbolic state $(l, \mathcal{Z})$. Assuming that the e-successor set, $Post_e(l, \mathcal{Z})$, was expressible as a single priced symbolic state $(l', \mathcal{Z}')$, this would obviously require $l' = m$ and $\mathcal{Z}' = (Z', c', r')$ with $Z' = \{y\}Z$. Furthermore, following our framework of Section 2, the cost-assignment of $\mathcal{Z}'$ should be such that $\mathsf{Cost}(u', \mathcal{Z}') = inf\{\mathsf{Cost}(u, \mathcal{Z}) \mid u \in \mathcal{Z} \wedge u[y \mapsto 0] = u'\}$ for all $u' \in \mathcal{Z}'$. Since $r(y) > 0$, it is obvious that these infima are obtained along the lower boundary of Z with respect to y (see Fig. 3 left). E.g. $\mathsf{Cost}((2, 0), \mathcal{Z}') = 4$, $\mathsf{Cost}((4, 0), \mathcal{Z}') = 2$, and $\mathsf{Cost}((6, 0), \mathcal{Z}') = 2$. In general $\mathsf{Cost}((x, 0), \mathcal{Z}') = \mathsf{Cost}((x, 2), \mathcal{Z}) = 6 - x$ for $2 \leq x \leq 5$ and $\mathsf{Cost}((x, 0), \mathcal{Z}') = \mathsf{Cost}((x, x - 3), \mathcal{Z}) = x - 4$ for $5 \leq x \leq 7$. However, the disagreement w.r.t. the cost-rate of x (-1 or 1) makes it clear that the desired cost-assignment is *not* linear and hence not obtainable from any *single* priced zone. On the other hand, it is also shows that splitting $Z' = \{y\}Z$ into the sub-zones $Z'_1 = Z' \wedge 2 \leq x \leq 5$ and $Z'_2 = Z' \wedge 5 \leq x \leq 7$, allows the e-successor set $Post_e(l, \mathcal{Z})$ to be expressed using the union of *two* priced zones (with $r(x) = -1$ in Z'_1 and $r(x) = 1$ in Z'_2).

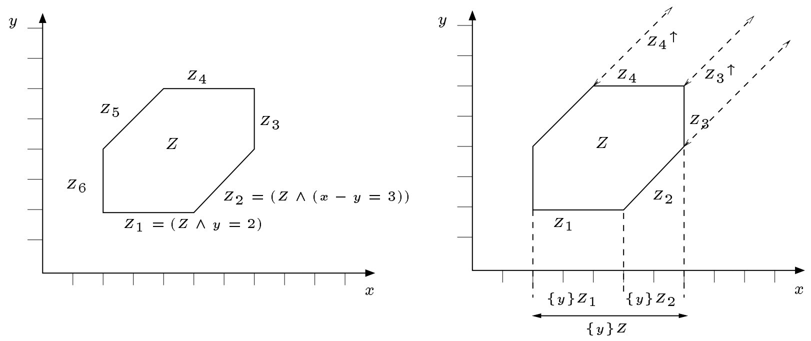

Fig. 4. A Zone: Facets and Operations.

Similarly, priced symbolic states are *not* directly closed w.r.t. $Post_\delta$. To see this, consider again the LPTA $\mathcal{A}$ from above and the priced zone $\mathcal{Z} = (Z, c, r)$ depicted in Fig. 3. Clearly, the set $Post_\delta(l, \mathcal{Z})$ must cover the zone $Z^\uparrow$ (see Fig. 3). It can be seen that, although $Post_\delta(l, \mathcal{Z})$ is not expressible as a *single* priced symbolic state, it may be expressed as a *finite union* by splitting the zone $Z^\uparrow$ into the three sub-zones Z, $Z_1^\uparrow = (Z^\uparrow \backslash Z) \wedge (x - y \leq 1)$, and $Z_2^\uparrow = (Z^\uparrow \backslash Z) \wedge (x - y \geq 1)$.

5 Facets & Operations on Priced Zones

The universal key to expressing successor sets of priced symbolic states as finite unions is provided by the notion of *facets* of a zone Z. Formally, whenever $x \bowtie n$ ($x - y \bowtie m$) is a constraint of Z, the strengthened zone $Z \wedge (x = n)$ ($Z \wedge (x - y = m)$) is a facet of Z. Facets derived from lower bounds on individual clocks, $x \geq n$, are classified as *lower facets*, and we denote by $LF(Z)$ the collection of all lower facets of Z. Similarly, the collection of *upper facets*, $UF(Z)$, of a zone Z is derived from upper bounds of Z. We refer to lower as well as upper facets as *individual clock* facets. Facets derived from lower bounds of the forms $x \geq n$ or $x - y \geq m$ are classified as lower *relative facets* w.r.t. x. The collection of lower relative facets of Z w.r.t. x is denoted $LF_x(Z)$. The collection of upper relative facets of Z w.r.t. x, $UF_x(Z)$, is derived similarly. Figure 4(left) illustrates a zone Z together with its six facets: e.g. $\{Z_1, Z_6\}$ constitutes the lower facets of Z, and $\{Z_1, Z_2\}$ constitutes the lower relative facets of Z w.r.t. y.

The importance of facets comes from the fact that they allow for decompositions of the delay- and reset-operations on zones as follows:

Lemma 1. *Let Z be a zone and y a clock. Then the following holds:*

i) $Z^\uparrow = \bigcup_{F \in LF(Z)} F^\uparrow$	*iii)* $\{y\}Z = \bigcup_{F \in LF_y(Z)} \{y\}F$
ii) $Z^\uparrow = Z \cup \bigcup_{F \in UF(Z)} F^\uparrow$	*iv)* $\{y\}Z = \bigcup_{F \in UF_y(Z)} \{y\}F$

Informally (see Fig. 4(right)) *i)* and *ii)* express that any valuation reachable by delay from Z is reachable from one of the lower facets of Z, as well as reachable from one of the upper facets of Z or within Z. *iii)* (and *iv)*) expresses that any valuation in the projection of a zone will be in the projection of the lower (upper) facets of the zone relative to the relevant clock.

As a first step, the delay- and reset-operation may be extended in a straightforward manner to priced (relative) facets:

Definition 3. *Let $\mathcal{Z} = (F, c, r)$ be a priced zone, where F is a relative facet w.r.t. y in the sense that $y - x = m$ is a constraint of F. Then $\{y\}\mathcal{Z} = (F', c', r')$, where $F' = \{y\}F$, $c' = c$, and $r'(x) = r(y) + r(x)$ and $r'(z) = r(z)$ for $z \neq x$. In case $y = n$ is a constraint of F, $\{y\}\mathcal{Z} = (F', c, r)$ with $F' = \{y\}F$.*[3]

Definition 4. *Let $\mathcal{Z} = (F, c, r)$ be a priced zone, where F is a lower or upper facet in the sense that $y = n$ is a constraint of F. Let $p \in \mathbb{N}$ be a cost-rate. Then $\mathcal{Z}^{\uparrow p} = (F', c', r')$, where $F' = F^{\uparrow}$, $c' = c$, and $r'(y) = p - \sum_{z \neq y} r(z)$ and $r'(z) = r(z)$ for $z \neq y$.*

Conjunction of constraints may be lifted from zones to priced zones simply by taking into account the possible change of the offset. Formally, let $\mathcal{Z} = (Z, c, r)$ be a priced zone and let $g \in \mathcal{B}(\mathbb{C})$. Then $\mathcal{Z} \wedge g$ is the priced zone $\mathcal{Z}' = (Z', c', r')$ with $Z' = Z \wedge g$, $r' = r$, and $c' = \mathsf{Cost}(\Delta_{Z'}, \mathcal{Z})$. For $\mathcal{Z} = (Z, c, r)$ and $n \in \mathbb{N}$ we denote by $\mathcal{Z} + n$ the priced zone $(Z, c + n, r)$.

The constructs of Definitions 3 and 4 essentially provide the *Post*-operations for priced facets. More precisely, it is easy to show that:

$$Post_e(l, \mathcal{Z}_1) = (l', \{y\}(\mathcal{Z}_1 \wedge g) + P(e)) \qquad Post_\delta(l, \mathcal{Z}_2) = (l, (\mathcal{Z}_2 \wedge I(l))^{\uparrow P(l)} \wedge I(l))$$

if $e = (l, g, \{y\}, l')$, $\mathcal{Z}_1$ is a priced relative facet w.r.t. to y and $\mathcal{Z}_2$ is an individual clock facet. Now, the following extension of Lemma 1 to priced symbolic states provides the basis for the effective realisation of *Post*-operations in general:

Theorem 1. *Let $\mathcal{A} = (L, l_0, E, I, P)$ be an LPTA. Let $e = (l, g, \{y\}, l') \in E$*[4] *with $P(e) = q$, $P(l) = p$, $I(l) = J$ and let $\mathcal{Z} = (Z, c, r)$ be a priced zone. Then:*

$$Post_e((l, \mathcal{Z})) = \begin{cases} \{\, (l', \{y\}Q + q) \mid Q \in LF_y(\mathcal{Z} \wedge g) \,\} & \text{if } r(y) \geq 0 \\ \{\, (l', \{y\}Q + q) \mid Q \in UF_y(\mathcal{Z} \wedge g) \,\} & \text{if } r(y) \leq 0 \end{cases}$$

$$Post_\delta((l, \mathcal{Z})) = \begin{cases} \{(l, \mathcal{Z})\} \;\cup\; \{\, (l, Q^{\uparrow p} \wedge J) \mid Q \in UF(\mathcal{Z} \wedge J)\} & \text{if } p \geq \sum_{x \in \mathbb{C}} r(x) \\ \{\, (l, Q^{\uparrow p} \wedge J) \mid Q \in LF(\mathcal{Z} \wedge J)\} & \text{if } p \leq \sum_{x \in \mathbb{C}} r(x) \end{cases}$$

In the definition of $Post_e$ the successor set is described as a union of either lower or upper relative facets w.r.t. to the clock y being reset, depending on the rate of y (as this will determine whether the minimum is obtained at the lower of

[3] This "definition" of $\{y\}(\mathcal{Z})$ is somewhat ambigious since it depends on which constraint involving y that is choosen. However, the Cost-function determined will be independent of this choice.

[4] For the case with a general reset-set r, the notion of relative facets may be generalized to sets of clocks.

upper boundary). For similar reason, in the definition of $Post_\delta$, the successor-set is expressed as a union over either lower or upper (individual clock) facets depending on the rate of the location compared to the sum of clock cost-rates.

To complete the instantiation of the framework of Section 2, it remains to indicate how to compute $minCost$ and $\sqsubseteq$ on priced symbolic states. Let $\mathcal{Z} = (Z, c, r)$ and $\mathcal{Z}' = (Z', c', r')$ be priced zones and let $(l, \mathcal{Z})$ and $(l', \mathcal{Z}')$ be corresponding priced symbolic states. Then $minCost(l, \mathcal{Z})$ is obtained by minimizing the linear expression $c + \sum_{x \in \mathbb{C}} (r(x) \cdot (x - \Delta_Z(x))$ under the (linear) constraints expressed by Z. Thus, computing $minCost$ reduces to solving a simple Linear Programming problem. Now let $\mathcal{Z}' \backslash \mathcal{Z}$ be the priced zone (Z^*, c^*, r^*) with $Z^* = Z$, $c^* = c' - \mathsf{Cost}(\Delta_{Z'}, \mathcal{Z})$ and $r^*(x) = r'(x) - r(x)$ for all $x \in \mathbb{C}$. It is easy to see that $\mathsf{Cost}(u, \mathcal{Z}' \backslash \mathcal{Z}) = \mathsf{Cost}(u, \mathcal{Z}') - \mathsf{Cost}(u, \mathcal{Z})$ for all $u \in \mathcal{Z}'$, and hence that $(l, \mathcal{Z}) \sqsubseteq (l', \mathcal{Z}')$ iff $l = l'$, $Z' \subseteq Z$ and $minCost(\mathcal{Z}' \backslash \mathcal{Z}) \geq 0)$ Thus, deciding $\sqsubseteq$ also reduces to a Linear Programming problem.

In exploring LPTAs using the algorithm of Fig. 1, we will only need to consider priced zones $\mathcal{Z}$ with non-negative cost assignments in the sense that $\mathsf{Cost}(u, \mathcal{Z}) \geq 0$ for all $u \in \mathcal{Z}$. Now, application of Higman's Lemma [Hig52] ensures that $\sqsubseteq$ is a well-quasi ordering on priced symbolic states for bounded LPTA. We refer to [BFH$^+$01] for more detailed arguments.

6 Implementation & Experiments

In this section we give further details on a prototype implementation within the tool Uppaal [LPY97] of priced zones, formally defined in the previous sections, and report on experiments on the aircraft landing problem.

The prototype implements the $Post_e$ (reset), $Post_\delta$ (delay), $minCost$, and $\sqsubseteq$ operations, using extensions of the DBM algorithms outlined in [Rok93]. To minimize the number of facets considered and reduce the size of the LP problems needed to be solved, we make heavy use of the canonical representation of zones in terms a *minimal* set of constraints given in [LLPY97]. For dealing with LP problems, our prototype currently uses a free available implementation of the simplex algorithm.[5] Many of the techniques for pruning and guiding the state space search described in [BFH$^+$] have been used extensively in modelling and verification.

Recall the aircraft landing problem partially described in Example 1. An LPTA model of the costs associated with landing a single aircraft is shown in Fig. 2(b). When landing several planes the schedule has to take into account the separation times between planes to avoid that the turbulence of one plane affecting an other. The separation times depend on the types of the planes that are involved. Large aircrafts for example generate more turbulence than small ones, and successive planes should consequently keep a bigger distance. To model the separation times between two types of planes we introduce an LPTA of the kind shown in Fig. 2(c).

[5] lp_solve 3.1a by Michael Berkelaar, `ftp://ftp.es.ele.tue.nl/pub/lp_solve`.

Table 1. Results for seven instances of the aircraft landing problem. Results were obtained on a PentiumII 333Mhz.

run-ways	problem instance	1	2	3	4	5	6	7
	number of planes	10	15	20	20	20	30	44
	number of types	2	2	2	2	2	4	2
	optimal value	700	1480	820	2520	3100	24442	1550
1	explored states	481	2149	920	5693	15069	122	662
	cputime (secs)	4.19	25.30	11.05	87.67	220.22	0.60	4.27
	optimal value	90	210	60	640	650	554	0
2	explored states	1218	1797	669	28821	47993	9035	92
	cputime (secs)	17.87	39.92	11.02	755.84	1085.08	123.72	1.06
	optimal value	0	0	0	130	170	0	
3	explored states	24	46	84	207715	189602	62	N/A
	cputime (secs)	0.36	0.70	1.71	14786.19	12461.47	0.68	
	optimal value				0	0		
4	explored states	N/A	N/A	N/A	65	64	N/A	N/A
	cputime (secs)				1.97	1.53		

Table 1 presents the results of an experiment were the prototype was applied to seven instances of the aircraft landing problem taken from [BKA00][6]. For each instance, which varies in the number of planes and plane types, we compute the cost of the optimal schedule. In cases the cost is non-zero we increase the number of runways until a schedule of cost 0 is found[7]. In all instances, the state space is explored in minimal cost-order, i.e. we select from the waiting list the priced zone $(l, \mathcal{Z})$ with lowest $minCost(l, \mathcal{Z})$. Equal values are distinguished by selecting first the zone which results from the largest number of transitions, and secondly by selecting the zone which involves the plane with the shortest target time. As can be seen from the table, our current prototype implementation is able to deal with all the tested instances. Beasley et al. [BKA00] solves all problem instances with a linear programming based tree search algorithm, in cases that the initial solution – obtained with a heuristic – is not zero. In 7 of the 15 benchmarks (with optimal solution greater than zero) the time-performance of our method is better than theirs. These are the instances 4 to 7 with less than 3 runways. This result also holds if we take into account that our computer is about 50% faster (according to the Dongarra Linpack benchmarks [Don01]). It should be noted, however, that our solution-times are quite incomparable to those of Beasleys. For some instances our approach is up to 25 times slower, while for others it is up to 50 times faster than the approach in [BKA00].

The cost-extended version of UPPAAL has additionally been (and is currently being) applied to other examples, including a cost-extended version of the Bridge Problem [RB98], an optimal broadcast problem and a testing problem.

[6] These and other benchmarks are available at `ftp://mscmga.ms.ic.ac.uk/pub/`.

[7] This is always possible as the cost of landing on target time is 0 and the number of runways can be increased until all planes arrive at target time.

7 Conclusion

In this paper we have considered the minimum-cost reachability problem for LPTAs. The notions of priced zones, and facets of a zone are central contributions of the paper underlying our extension of the tool UPPAAL. Our initial experimental investigations based on a number of examples are quite encouraging.

Compared with the existing special-purpose, time-optimizing version of UPPAAL [BFH+], the presented general cost-minimizing implementation does only marginally down-grade performance. In particular, the theoretical possibility of uncontrolled splitting of zones does not occur in practice. In addition, the consideration of non-uniform cost seems to significantly reduce the number of symbolic states explored.

The single, most important question, which calls for future research, is how to exploit the simple structure of the LP-problems considered. We may benefit significantly from replacing the currently used LP package with some package more tailored towards small-size problems.

References

[AC91] D. Applegate and W. Cook. A Computational Study of the Job-Shop Scheduling Problem. *OSRA Journal on Computing 3*, pages 149–156, 1991.

[ACJYK96] P. Abdulla, K. Cerans, B. Jonsson, and T. Yih-Kuen. General decidability theorems for infinite-state systems, 1996.

[AD90] R. Alur and D. Dill. Automata for Modelling Real-Time Systems. In *Proc. of Int. Colloquium on Algorithms, Languages and Programming*, number 443 in Lecture Notes in Computer Science, pages 322–335, July 1990.

[AJ94] P. Abdulla and B. Jonsson. Undecidability of verifying programs with unreliable channels. In *Proc. 21st Int. Coll. Automata, Languages, and Programming (ICALP'94)*, volume 820 of *LNCS*, 1994.

[ATP] R. Alun, S. La Torre, and G. J. Pappas. Optimal paths in weighted timed automata. To appear in HSCC2001.

[BDM+98] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, and S. Yovine. Kronos: A Model-Checking Tool for Real-Time Systems. In *Proc. of the 10th Int. Conf. on Computer Aided Verification*, number 1427 in Lecture Notes in Computer Science, pages 546–550. Springer–Verlag, 1998.

[BFH+] G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, and J. Romijn. Efficient guiding towards cost-optimality in UPPAAL. To appear in Proceedings of TACAS'2001.

[BFH+01] G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. Vaandrager. Minimum-Cost Reachability for Priced Timed Automata. To appear in Proceedings of HSCC2001, 2001.

[BKA00] J.E. Beasley, M. Krishnamoorthy, and D. Abramson. Scheduling Aircraft Landings-The Static Case. *Transportation Science*, 34(2):180–197, 2000.

[BM00] Ed Brinksma and Angelika Mader. Verification and optimization of a plc control schedule. In *Proceedings of the 7th SPIN Workshop*, volume 1885 of *Lecture Notes in Computer Science*. Springer Verlag, 2000.

[Cer94] K. Cerans. Deciding properties of integral relational automata. In *Proceedings of ICALP 94*, volume 820 of *LNCS*, 1994.

[Dil89] D. Dill. Timing Assumptions and Verification of Finite-State Concurrent Systems. In J. Sifakis, editor, *Proc. of Automatic Verification Methods for Finite State Systems*, number 407 in Lecture Notes in Computer Science, pages 197–212. Springer–Verlag, 1989.

[Don01] Jack J. Dongarra. Performance of Various Computers Using Standard Linear Equations Software. Technical Report CS-89-85, Computer Science Department, University of Tennessee, 2001. An up-to-date version of this report can be found at `http://www.netlib.org/benchmark/performance.ps`.

[Feh99] A. Fehnker. Scheduling a steel plant with timed automata. In *Proceedings of the 6th International Conference on Real-Time Computing Systems and Applications (RTCSA99)*, pages 280–286. IEEE Computer Society, 1999.

[FS98] A. Finkel and P. Schnoebelen. Fundamental structures in well-structured infinite transition systems. In *Proc. 3rd Latin American Theoretical Informatics Symposium (LATIN'98)*, volume 1380 of *LNCS*, 1998.

[FS01] A. Finkel and Ph. Schnoebelen. Well structured transition systems everywhere. *Theoretical Computer Science*, 256(1-2):64–92, 2001.

[HHWT97] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. HyTech: A Model Checker for Hybird Systems. In Orna Grumberg, editor, *Proc. of the 9th Int. Conf. on Computer Aided Verification*, number 1254 in Lecture Notes in Computer Science, pages 460–463. Springer–Verlag, 1997.

[Hig52] G. Higman. Ordering by divisibility in abstract algebras. *Proc. of the London Math. Soc.*, 2:326–336, 1952.

[HLP00] T. Hune, K. G. Larsen, and P. Pettersson. Guided Synthesis of Control Programs Using Uppaal. In Ten H. Lai, editor, *Proc. of the IEEE ICDCS International Workshop on Distributed Systems Verification and Validation*, pages E15–E22. IEEE Computer Society Press, April 2000.

[LLPY97] Fredrik Larsson, Kim G. Larsen, Paul Pettersson, and Wang Yi. Efficient Verification of Real-Time Systems: Compact Data Structures and State-Space Reduction. In *Proc. of the 18th IEEE Real-Time Systems Symposium*, pages 14–24. IEEE Computer Society Press, December 1997.

[LPY97] K. G. Larsen, P. Pettersson, and W. Yi. Uppaal in a Nutshell. *Int. Journal on Software Tools for Technology Transfer*, 1(1–2):134–152, October 1997.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice Hall, Englewood Cliffs, 1989.

[NY99] P. Niebert and S. Yovine. Computing optimal operation schemes for multi batch operation of chemical plants. VHS deliverable, May 1999. Draft.

[RB98] T. C. Ruys and E. Brinksma. Experience with Literate Programming in the Modelling and Validation of Systems. In Bernhard Steffen, editor, *Proceedings of the Fourth International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'98)*, number 1384 in Lecture Notes in Computer Science (LNCS), pages 393–408, Lisbon, Portugal, April 1998. Springer-Verlag, Berlin.

[Rok93] T. G. Rokicki. *Representing and Modeling Digital Circuits.* PhD thesis, Stanford University, 1993.