Research Article

AND DATA SCIENCES
BULLETIN OF COMPUTER

Cost-Optimal Reachability for Timed Automata with Diagonals via Zone Simulations

Sayan Mukherjee

Chennai Mathematical Institute, Chennai, India

Abstract

We study cost-optimal reachability in priced timed automata whose guards and invariants may include diagonal constraints of the form $x - y \sim c$. Classical extrapolation and zone-splitting approaches struggle with diagonals, often incurring exponential blow-ups or losing optimality guarantees. We propose a simulation-based priced zone exploration that (i) keeps *actual* zones with diagonals, (ii) propagates sound lower bounds on accumulated cost, and (iii) prunes the search using an antichain partial order combining zone simulation with cost dominance. We derive admissible A*-style heuristics from dynamic LU bounds generalized to diagonals, and we prove soundness and conditional termination for subclasses where the induced abstraction has finitely many antichain minima. An implementation evaluates diagonal-heavy scheduling and protocol benchmarks against diagonal-free conversion and state-of-the-art priced engines. Experiments show consistent reductions in explored nodes and time while preserving optimal costs, enabling cost-optimal analysis on models previously intractable under diagonal splitting.

Keywords: priced timed automata, cost-optimal reachability, diagonal constraints, simulation-based zone exploration

1. Introduction

Timed automata (TA) are a standard mathematical model for designing and verifying real-time systems. In many practical settings, quantitative relations between clocks are essential: offsets, precedence constraints, headway limits in transportation and scheduling, or bounded inter-arrival times in communication protocols naturally give rise to *diagonal constraints* of the form $x - y \sim c$, where $\sim \in \{<, \leq, =, \geq, >\}$ and $c \in \mathbb{Z}$ [1]. While such constraints increase modeling fidelity, they also make symbolic state-space exploration substantially harder. Eliminating diagonals by introducing additional clocks or duplicating locations can preserve semantics but typically causes large blow-ups in the number of clocks and symbolic states, hampering performance and obscuring diagnostic traces. Keeping diagonals with extrapolation and zone-splitting fares no better: extrapolations that are sound and effective for non-diagonal guards can become unsound or overly conservative once differences $x - y$ are present, and splitting a zone along each diagonal guard may trigger exponential fragmentation [2].

The situation is even more delicate for *priced* timed automata (PTA), in which locations accrue cost at nonnegative rates and edges carry discrete weights. Here the objective is not merely to decide

Corresponding author: Sayan Mukherjee (sayanm@cmi.ac.in)

reachability but to compute the *optimal* (minimum) accumulated cost to a target. Heuristics that discard timing precision to gain speed can invalidate optimality guarantees, whereas naive exact methods are easily overwhelmed by the combinatorics introduced by diagonal constraints [3–5]. As a result, cost-optimal analysis for diagonal-heavy models—such as offset-based schedulers, skew-tolerant mutual exclusion protocols, and headway control systems—remains challenging [6].

These difficulties with a *simulation-based* exploration for cost-optimal reachability that avoids diagonal elimination and zone-splitting altogether [7, 8]. The method retains *actual* diagonal zones represented directly as difference-bound matrices over clock differences, augments each symbolic node with a provably sound *lower bound* on the best achievable cost from the initial configuration, and prunes the frontier using an *antichain* dominance relation that couples zone simulation with *cost dominance* [8–13]. The search is guided by admissible A*-style heuristics derived from *diagonal-aware LU* summaries and invariant slack, and it can operate in an *anytime* fashion: even when termination is not guaranteed, the algorithm maintains improving upper bounds together with certified suboptimality gaps [14–16].

This work targets core problems in formal verification and real-time synthesis and has immediate implications for optimal scheduling, protocol design, and planning under timing constraints. The techniques integrate cleanly with heuristic search (A*, Weighted-A*) and complement existing priced-zone engines by restoring diagonal precision without prohibitive splitting, thereby enabling tractable cost-optimal analysis on models that were previously out of reach.

2. Preliminaries

We recall standard notions for timed automata with costs and make explicit the use of diagonal constraints. Throughout, let $X = \{x_1, \dots, x_n\}$ be a finite set of clocks and let $X_0 = X \cup \{x_0\}$ with a distinguished dummy clock x_0 fixed to 0. A *valuation* is a map $v : X \to \mathbb{R}_{\geq 0}$, extended with $v(x_0) = 0$. For $d \in \mathbb{R}_{\geq 0}$, write $v + d$ for the valuation defined by $(v + d)(x) = v(x) + d$ for all $x \in X$.

A (closed or strict) *difference constraint* has the form

$$x_i - x_j \ \sim \ c \qquad (x_i, x_j \in X_0,\ \sim \in \{<, \leq, =, \geq, >\},\ c \in \mathbb{Z}),$$

and a *guard* is a finite conjunction of such constraints; an *invariant* is defined likewise. Constraints with $x_j = x_0$ encode simple bounds $x_i \sim c$, whereas constraints with $x_i \neq x_j$ are *diagonal constraints*. We write $v \models g$ if valuation v satisfies guard g.

Definition 1 (Priced Timed Automaton with Diagonals)**.** A PTA is a tuple $\mathcal{A} = (L, \ell_0, X, E, \mathrm{Inv}, \mathrm{Rate}, w)$ where:

- L is a finite set of locations with initial $\ell_0 \in L$;
- X is a finite set of clocks;
- $E \subseteq L \times G(X) \times 2^X \times L$ is a finite set of edges $e = (\ell, g, R, \ell')$ with guard g, reset set $R \subseteq X$, and target ℓ';
- $\mathrm{Inv} : L \to G(X)$ assigns an invariant to each location (diagonals allowed);
- $\mathrm{Rate} : L \to \mathbb{R}_{\geq 0}$ assigns a nonnegative cost rate to each location;

- $w : E \to \mathbb{R}_{\geq 0}$ assigns a nonnegative discrete cost to each edge.

The semantics is a labelled transition system over states $(\ell, v) \in L \times \mathbb{R}^X_{\geq 0}$ with two kinds of transitions:

- Delay $(\ell, v) \xrightarrow{d} (\ell, v + d)$ for $d \in \mathbb{R}_{\geq 0}$ iff $v + \tau \models \mathrm{Inv}(\ell)$ for all $\tau \in [0, d]$. The *delay cost* accrued is $\mathrm{Rate}(\ell) \cdot d$.
- Discrete $(\ell, v) \xrightarrow{e} (\ell', v')$ for $e = (\ell, g, R, \ell') \in E$ iff $v \models g$ and

$$v' = v[R \leftarrow 0] \quad (\text{i.e. } v'(x) = 0 \text{ for } x \in R,\ v'(x) = v(x) \text{ otherwise}) \quad \text{and} \quad v' \models \mathrm{Inv}(\ell').$$

  The *edge cost* accrued is $w(e)$.

A (finite) *run* is a sequence alternating delays and discrete steps that respects the above rules. Its total cost is the sum of all delay and edge costs. Given a set of target locations $T \subseteq L$, the *cost-optimal reachability* value is

$$\mathrm{Cost}(\mathcal{A}, T) \;=\; \inf\{\, \mathrm{Cost}(\rho) \mid \rho \text{ is a run from } (\ell_0, \mathbf{0}) \text{ to some } (\ell, v) \text{ with } \ell \in T \,\}.$$

Because rates and edge costs are nonnegative, Cost is well-defined; when the infimum is not achieved, we can still speak of ε-optimal runs for any $\varepsilon > 0$.

A *zone* $Z \subseteq \mathbb{R}^X_{\geq 0}$ is a convex set definable by a conjunction of difference constraints $x_i - x_j \sim c$. Zones are represented using *difference-bound matrices* (DBMs) indexed by X_0: an entry $M[i, j]$ stores the tightest known bound $x_i - x_j \sim c$. The canonical (closed) form of a DBM is obtained by all-pairs shortest paths over the induced constraint graph; emptiness corresponds to the presence of a negative cycle (for $\leq$ bounds) or an incompatible strictness. Standard operations used in symbolic exploration are:

- Intersection with a guard/invariant $Z \cap g$, implemented by tightening DBM entries for the constraints in g followed by canonicalization.
- Time-elapse (also called *up-closure*) $Z^{\to}$, which relaxes upper bounds on clocks to model arbitrary nonnegative delay while staying within the invariant of the current location.
- Reset $Z[R \leftarrow 0]$, which sets clocks in R to zero by updating the corresponding rows/columns relative to x_0 and re-closing.

All operations run in $O(|X|^3)$ time due to the closure step, and preserve convexity. Diagonal constraints $x_i - x_j \sim c$ are *first-class* in DBMs: no extra clocks are introduced, unlike diagonal-elimination translations which may blow up the model size.

At the symbolic level we manipulate pairs (ℓ, Z) with Z a zone of valuations satisfying $\mathrm{Inv}(\ell)$. The standard post operators are

$$\begin{aligned} \text{Delay:} \quad \mathsf{Post}_\delta(\ell, Z) &= (\ell,\ (Z \cap \mathrm{Inv}(\ell))^{\to}), \\ \text{Edge:} \quad \mathsf{Post}_e(\ell, Z) &= (\ell',\ ((Z \cap g)[R \leftarrow 0]) \cap \mathrm{Inv}(\ell')), \end{aligned}$$

for $e = (\ell, g, R, \ell') \in E$. These two operators generate the symbolic reachability graph explored by our algorithm.

Given a symbolic state (ℓ, Z), delaying by $d \geq 0$ within $\mathrm{Inv}(\ell)$ contributes $\mathrm{Rate}(\ell) \cdot d$ to the cost; taking edge $e = (\ell, g, R, \ell')$ adds $w(e)$. In later sections we attach to each symbolic node a scalar lower bound on the minimal accumulated cost to reach it; here we only fix notation for the underlying timed-cost semantics.

We consider target sets $T \subseteq L$ (reach any valuation in a target location). For heuristic purposes we also use the *location graph* $(L, \{(\ell, \ell') \mid \exists e = (\ell, g, R, \ell')\})$ and its shortest *discrete* path costs with edge weights $w(e)$ (ignoring time), yielding admissible baseline estimates in the search that follows.

This formalization aligns with the goals stated in the introduction: diagonals are modeled natively in guards and invariants, symbolic successors are computed without zone-splitting, and DBMs over differences provide the algebraic substrate for efficient intersection, delay, and reset while preserving precision necessary for cost-optimal analysis.

3. Simulation-Based Priced Exploration

We explore a symbolic graph whose nodes carry both timing and cost information and are manipulated without diagonal elimination or zone-splitting. The exploration proceeds by repeatedly computing post-images of symbolic states and maintaining certified *lower* and *upper* bounds on the optimal cost. Lower bounds drive the search order and pruning; upper bounds enable an anytime mode with guaranteed suboptimality gaps [17].

3.1. Search state and initialization

A symbolic state is a tuple

$$s = (\ell, Z, \mathrm{LB}, \mathrm{UB}),$$

where $\ell \in L$, Z is an *actual* diagonal zone (a DBM over differences) such that $Z \subseteq \mathrm{Inv}(\ell)$, $\mathrm{LB} \in \mathbb{R}_{\geq 0}$ is a certified lower bound on the optimal cost to reach (ℓ, Z) from the initial state, and $\mathrm{UB} \in \mathbb{R}_{>0} \cup \{\infty\}$ stores the best known *feasible* cost to reach any target via a concrete run going through this node (used by the anytime mode; it may be omitted at construction time). The exploration is seeded with

$$s_0 = (\ell_0,\ Z_0,\ 0,\ \infty), \qquad Z_0 = \{\mathbf{0}\} \cap \mathrm{Inv}(\ell_0),$$

and a global incumbent $UB^\star \leftarrow \infty$.

3.2. Symbolic successors

We compute successors by the standard post operators (defined in the preliminaries) while *retaining* diagonal constraints in the DBM. For a nonempty state (ℓ, Z),

$$\begin{aligned} \textbf{Delay:} \quad & \mathsf{Post}_\delta(\ell, Z) \ = \ (\ell,\ (Z \cap \mathrm{Inv}(\ell))^{\rightarrow}), \\ \textbf{Edge:} \quad & \mathsf{Post}_e(\ell, Z) \ = \ (\ell',\ ((Z \cap g)[R \leftarrow 0]) \cap \mathrm{Inv}(\ell')), \end{aligned}$$

for $e = (\ell, g, R, \ell') \in E$. We discard empty zones after each operation. Delay transitions accrue continuous cost at rate $\mathrm{Rate}(\ell)$; edge transitions accrue discrete cost $w(e)$.

3.3. Lower-bound propagation (pathwise)

Each expansion updates a pathwise lower bound by adding a *step lower bound* $\underline{c}_{\mathrm{step}}$:

$$\mathrm{LB}' \ = \ \mathrm{LB} \ + \ \underline{c}_{\mathrm{step}}((\ell, Z) \to (\ell', Z')).$$

We use diagonal-aware formulas that are efficient and safe:

(i) Edge steps. For $e = (\ell, g, R, \ell')$,

$$\underline{c}^{\text{edge}}_{\text{step}} \;=\; w(e),$$

because the semantic edge itself does not require time elapse; any delay cost appears on preceding delay steps.

(ii) Delay steps (baseline). A universally safe choice is

$$\underline{c}^{\text{delay}}_{\text{step}} \;=\; 0,$$

which preserves admissibility but carries no information. We strengthen it using *invariant/guard slack* as follows.

(iii) Delay steps (invariant-guard enabling). Fix outgoing edges $E(\ell) = \{(\ell, g_i, R_i, \ell'_i)\}$. For a guard g, define the minimal enabling delay

$$d_{\min}(Z, g) \;=\; \inf\{\, d \geq 0 \mid ((Z \cap \text{Inv}(\ell))^{\uparrow d}) \cap g \neq \emptyset \,\},$$

where $(\cdot)^{\uparrow d}$ denotes exact time elapse by d while remaining in $\text{Inv}(\ell)$. Then

$$\underline{c}^{\text{delay}}_{\text{step}} \;\geq\; \text{Rate}(\ell) \cdot \max\!\big(0,\; \min_{e \in E(\ell)} d_{\min}(Z, g_e)\big).$$

Crucially, diagonal constraints of the form $x_i - x_j \sim c$ are *time-invariant* under elapse (both sides increase by the same d), so they do not reduce $d_{\min}$. Hence $d_{\min}$ can be computed from the *simple* (non-diagonal) literals in guards and the invariant, using DBM slack to threshold bounds. This yields a cheap yet often positive delay lower bound.

3.4. Monotonicity and admissibility

For any successor (ℓ', Z') produced above, the propagated value LB' satisfies

$$\text{LB}' \;\leq\; \text{OptCost}((\ell', Z')),$$

i.e., it never *overestimates* the true optimal cost to reach (ℓ', Z'). This follows because edge contributions are exact discrete costs and delay contributions use lower bounds on necessary time elapse, multiplied by nonnegative rates [18–20]. By induction along any explored path, the LB attached to a node is a valid lower bound on the optimal cost from the initial state to that node. These properties make LB suitable as the g-component in A*-style keys $f = \text{LB} + h$ with any admissible heuristic h (see §5).

3.5. Feasible upper bounds and anytime mode

Whenever we encounter a target state (ℓ, Z) with $\ell \in T$, we reconstruct a concrete run by selecting a witness valuation sequence (v_k) consistent with the nonempty intersections taken along the path (standard DBM extraction). The exact run cost C_{wit} provides a global incumbent

$$UB^\star \;\leftarrow\; \min\{\, UB^\star,\; C_{\text{wit}} \,\}.$$

We also set the node-local UB $\leftarrow UB^\star$ for bookkeeping. At any time during the search, we thus maintain a certified suboptimality gap

$$0 \;\leq\; UB^\star \;-\; \min_{s \in \text{frontier}} (\text{LB}(s) + h(s)),$$

which decreases as either the incumbent improves or frontier lower bounds tighten.

3.6. *Frontier discipline (preview)*

In the core exploration we keep a priority queue keyed by $f(s) = \mathrm{LB}(s) + h(s)$. To avoid redundant work, we combine this with an antichain-based dominance filter over zones and costs (formalized in §4). Intuitively, if a state s is timing-wise subsumed by s' and has no better lower bound, then expanding s cannot lead to a cheaper target than expanding s'.

3.7. *Termination and partial correctness*

If the diagonal-aware abstraction used by the dominance test induces only finitely many antichain minima per location, the exploration terminates and returns the exact optimum (see §6). Otherwise, the algorithm remains *anytime*: it never violates admissibility of LB and keeps improving $UB^\star$ as new witnesses are found, along with a certified optimality gap.

4. Antichain Pruning and Diagonal-Aware Abstraction

To keep the exploration finite without splitting zones, we generalize classical LU-abstractions to *diagonal* bounds. Intuitively, the abstraction α relaxes (clips) constraints that exceed precomputed per-difference limits, yielding a *superset* of the original zone that preserves reachability while merging many concretely distinct zones into a small number of abstract representatives [21–25].

For every ordered pair of clocks $(x, y) \in X_0^2$ we collect two nonnegative integers L_{xy}, U_{xy} (with the convention that $L_{xy} = U_{xy} = +\infty$ if no bound is known). They summarize all constants that appear in the model on differences $x - y$:

- U_{xy} upper-bounds literals of the form $x - y \le c$;
- L_{xy} upper-bounds literals of the form $y - x \le c$ (i.e., $x - y \ge -c$).

We obtain a *static* summary by scanning guards and invariants of $\mathcal{A}$. During exploration we further apply *dynamic learning*: whenever we encounter tighter guard/invariant constants, we non-decreasingly update (L_{xy}, U_{xy}). This strengthening only coarsens the abstraction (makes α larger), never endangering soundness.

Let Z be a closed DBM over X_0 with canonical matrix M_Z, where each entry $M_Z[i, j]$ encodes the tightest known bound $x_i - x_j \le c$ (with a strictness bit which we omit in notation). Define $\alpha(Z)$ as the zone represented by the DBM M_α whose entries are obtained by per-entry *clipping*:

$$M_\alpha[i,j] \;=\; \begin{cases} \min(M_Z[i,j],\ U_{x_i x_j}) & \text{(clip upper bounds)} \\ \text{and ensure } M_\alpha[j,i] \le L_{x_i x_j} & \text{(clip reverse bound = lower bound on } x_i - x_j). \end{cases}$$

Operationally, clipping an entry to a larger value means *relaxing* its constraint; if $M_Z[i, j] > U_{x_i x_j}$ we set $M_\alpha[i, j] := U_{x_i x_j}$, and if $-M_Z[j, i] > L_{x_i x_j}$ we set $M_\alpha[j, i] := L_{x_i x_j}$. After clipping we re-close the DBM. By construction,

$$Z \;\subseteq\; \alpha(Z) \quad \text{and} \quad \alpha \text{ is monotone: } Z \subseteq Z' \Rightarrow \alpha(Z) \subseteq \alpha(Z').$$

Strict inequalities are preserved unless the affected entry is relaxed; when a constraint is dropped (clipped to $+\infty$), strictness becomes irrelevant.

Lemma 2 (Post-compatibility). *For every location ℓ and zone $Z \subseteq \mathrm{Inv}(\ell)$, the abstraction is forward-simulation compatible:*

$$\alpha(\mathsf{Post}_\delta(\ell, Z)) \subseteq \mathsf{Post}_\delta(\ell, \alpha(Z)), \qquad \alpha(\mathsf{Post}_e(\ell, Z)) \subseteq \mathsf{Post}_e(\ell, \alpha(Z)),$$

for all edges e outgoing from ℓ.

Proof. α only *relaxes* constraints by clipping to (L_{xy}, U_{xy}). Relaxation cannot disable time-elapse within $\mathrm{Inv}(\ell)$ nor a guard g, hence any concrete successor valuation from Z is also a successor from $\alpha(Z)$. Closure preserves these inclusions at the DBM level. □

We order states at the same location by timing subsumption under α and by cost lower bounds.

Definition 3 (Dominance and Antichain). For states $s = (\ell, Z, \mathrm{LB})$ and $s' = (\ell', Z', \mathrm{LB}')$ with $\ell = \ell'$, we write

$$s' \preceq s \quad \text{iff} \quad Z \subseteq \alpha(Z') \quad \text{and} \quad \mathrm{LB}' \leq \mathrm{LB}.$$

A set $\mathcal{F}$ of states is an *antichain* if no two distinct states in $\mathcal{F}$ are comparable by $\preceq$.

Intuitively, if $s' \preceq s$, then s is timing-wise redundant (its zone is contained in the abstraction of s') and also no better in terms of already-accumulated cost.

Theorem 4 (Safe Pruning). *Let $s' \preceq s$ with $\ell(s') = \ell(s)$. Expanding s cannot yield a target with cost strictly smaller than what is achievable by expanding s'. Hence discarding s is sound for optimal-cost reachability.*

Proof. By Lemma 2, every symbolic successor of s is simulated by a successor of s' under α. Moreover, $\mathrm{LB}' \leq \mathrm{LB}$ implies that along any matched expansion, the pathwise lower bound from s' is never worse *at insertion time*. Because our search only uses admissible lower bounds and takes minima over explored paths, pruning s cannot remove the unique prefix of a strictly cheaper optimal run than what s' already permits. □

We maintain the open set as a *minimal antichain* under $\preceq$. When a new state t is generated, we perform:

1. Dominance test (discard): if $\exists u$ in the frontier with $u \preceq t$, drop t.
2. Prune dominated: otherwise, remove all u with $t \preceq u$ and insert t.

For zones represented by canonical DBMs, inclusion $Z \subseteq Z'$ is equivalent to the component-wise test $M_Z[i,j] \leq M_{Z'}[i,j]$ for all i, j, so step (1) and (2) reduce to $O(|X|^2)$ comparisons per candidate after a single $O(|X|^3)$ closure of $\alpha(Z)$. In practice we index frontier elements by hashes of $\alpha(Z)$ to narrow candidates before exact checks.

If, for each location ℓ, the set of abstract zones $\{\alpha(Z) \mid Z \subseteq \mathrm{Inv}(\ell)\}$ has only finitely many *minimal* elements under set inclusion (e.g., because all relevant difference constants are bounded by LU), then the antichain maintained by Algorithm 1 cannot grow unboundedly. This yields the conditional termination statement formalized in §6.

When clipping, we retain strictness for entries that remain finite; when a bound is relaxed (dropped), the strictness bit is ignored. Equality constraints $x - y = c$ are represented by the pair $x - y \leq c$ and $y - x \leq -c$; clipping one side may relax equality to an interval—this is safe since $\alpha(Z)$ is a superset of Z.

Algorithm 1 AntichainInsert(t) for frontier $\mathcal{F}$

Require: New state $t = (\ell, Z, \mathrm{LB})$; frontier $\mathcal{F} \subseteq \{\ell\} \times \mathcal{Z} \times \mathbb{R}_{\geq 0}$

for all $u = (\ell, Z_u, \mathrm{LB}_u) \in \mathcal{F}$ **do**
  if $Z \subseteq \alpha(Z_u)$ and $\mathrm{LB}_u \leq \mathrm{LB}$ **then**
    return DISCARD
for all $u = (\ell, Z_u, \mathrm{LB}_u) \in \mathcal{F}$ **do**
  if $Z_u \subseteq \alpha(Z)$ and $\mathrm{LB} \leq \mathrm{LB}_u$ **then**
    remove u from $\mathcal{F}$
insert t into $\mathcal{F}$; **return** INSERT

5. Heuristics and A* Search

We employ a best-first search where each open state s is keyed by

$$f(s) \;=\; \mathrm{LB}(s) + h(s),$$

with h an *admissible* estimate of the remaining optimal cost from s to any target. Lower bounds LB are pathwise and certified (cf. §3); heuristics h are diagonal-aware but cheap to compute. We make the notation explicit: write $s = (\ell, Z, \mathrm{LB}, \mathrm{UB})$ and let $T \subseteq L$ be the target locations.

Let the *location graph* be $G_L = (L, E_L)$ with $(\ell, \ell') \in E_L$ iff $\exists e = (\ell, g, R, \ell') \in E$. The *discrete weight* of (ℓ, ℓ') is

$$w^\star(\ell, \ell') \;=\; \min\{\, w(e) \mid e = (\ell, g, R, \ell') \in E \,\},$$

and the discrete distance to the target is

$$d_{\mathrm{disc}}(\ell) \;=\; \min_{\pi\,:\,\ell \rightsquigarrow T} \sum_{(\ell_i, \ell_{i+1}) \in \pi} w^\star(\ell_i, \ell_{i+1}),$$

with $d_{\mathrm{disc}}(\ell) = +\infty$ if T is unreachable in G_L. Since $w^\star \geq 0$, we can precompute d_{disc} by a multi-source Dijkstra from T.

Fix a location ℓ and zone $Z \subseteq \mathrm{Inv}(\ell)$. For each outgoing edge $e = (\ell, g, R, \ell')$, define the minimal enabling delay

$$d_{\min}(Z, g) \;=\; \inf \left\{\, d \geq 0 \;\middle|\; (Z \cap \mathrm{Inv}(\ell))^{\uparrow d} \cap g \neq \emptyset \,\right\},$$

where $X^{\uparrow d}$ denotes exact time elapse by d within $\mathrm{Inv}(\ell)$. Because diagonal literals $x - y \sim c$ are invariant under time elapse, only *simple* bounds constrain $d_{\min}$; it is computable from DBM slack in $O(|X|^2)$. A conservative local time-cost floor is then

$$h_{\mathrm{wait}}(\ell, Z) \;=\; \mathrm{Rate}(\ell) \cdot \min_{e \in \mathrm{out}(\ell)} d_{\min}(Z, g_e),$$

with the convention $\min \emptyset = 0$ for sinks.

We use three basic admissible heuristics and a combined one:

- Zero: $h_0(\ell, Z) = 0$.
- Discrete: $h_{\mathrm{disc}}(\ell, Z) = d_{\mathrm{disc}}(\ell)$. This lower-bounds all future *edge* costs while ignoring time.
- Rate-aware local: $h_{\mathrm{rate}}(\ell, Z) = h_{\mathrm{wait}}(\ell, Z)$. This lower-bounds the *immediate* delay cost required before any edge can fire.

- Combined (default): $h_{\oplus}(\ell, Z) = h_{\text{disc}}(\ell, Z) + h_{\text{rate}}(\ell, Z)$. Since the two components account for disjoint cost sources (discrete vs. the *initial* necessary waiting), their sum remains admissible.

Lemma 5 (Admissibility)**.** *For any state $s = (\ell, Z, \cdot, \cdot)$, we have $0 \le h_0(s) \le h_{\oplus}(s) \le \text{Cost}((\ell, Z) \rightsquigarrow T)$. In particular, $h_0, h_{disc}, h_{rate}, h_{\oplus}$ are admissible.*

Proof. h_0 is trivial. h_{disc} lower-bounds the sum of discrete costs along any run since every discrete step contributes at least the minimum edge weight on its location hop. h_{rate} lower-bounds the *first* delay cost because any enabled edge from Z requires at least $d_{\min}$ time elapse in ℓ, incurring $\text{Rate}(\ell) \cdot d_{\min}$. Adding these two non-overlapping lower bounds preserves admissibility. □

A heuristic h is *consistent* if $h(s) \le \underline{c}_{\text{step}}(s \to s') + h(s')$ for every successor s' of s. Consistency avoids re-openings in A*.

Lemma 6. *h_{disc} is consistent when edge weights are nonnegative; h_{rate} is consistent for delay successors and nonincreasing over edge successors; hence $h_{\oplus}$ is consistent.*

Proof. For edge steps, d_{disc} satisfies a Bellman inequality on G_L. For delays, d_{disc} does not increase. h_{rate} decreases by at most the delay we just paid; for edge steps it may drop to a new local floor (never increasing). Summation preserves consistency. □

5.1. Weighted A and ε-admissibility*

For faster anytime behavior we allow a weight $\omega \ge 1$ and use

$$f_{\omega}(s) = \text{LB}(s) + \omega\, h(s).$$

This preserves ω-suboptimality: the first target popped from the queue has cost at most ω times optimal. We default to $\omega = 1.0$ (exact A*) in the main experiments and report $\omega \in \{1.2, 1.5\}$ as a speed/quality trade-off.

5.2. Duplicate detection and re-insertion policy

We use the antichain frontier $\mathcal{F}$ (see §4) as the *closed* structure: a candidate s' is discarded if dominated by an existing element; otherwise we remove dominated elements and insert s'. With consistent heuristics (Lemma 6), nodes do not need to be re-opened; if consistency is disabled (e.g., with an aggressive learned heuristic), we reinsert when a strictly smaller LB is found.

5.3. Anytime bound and stopping rules

At all times we maintain a certified optimality gap

$$\text{gap} \;=\; UB^{\star} - \min_{s \in Q}(\text{LB}(s) + h(s)) \;\;\ge\;\; 0.$$

We can stop when gap $\le \varepsilon$ (user tolerance) or when the queue is empty (exact optimum found). With f_{ω}, the first goal popped is ω-admissible.

Algorithm 2 A* with Antichain Pruning for Priced TA with Diagonals

Require: Initial state $s_0 = (\ell_0, Z_0, \mathrm{LB} = 0, \mathrm{UB} = \infty)$, target set T, heuristic h, weight $\omega \geq 1$

$Q \leftarrow$ priority queue keyed by $f_\omega(s) = \mathrm{LB}(s) + \omega\, h(s)$; insert s_0
$\mathcal{F} \leftarrow \{s_0\}$ ▷ minimal antichain frontier
$UB^\star \leftarrow \infty$
while Q not empty **do**
  $s \leftarrow$ pop-min$Q$
  if $s.\ell \in T$ **then**
    $UB^\star \leftarrow \min(UB^\star, \mathrm{LB}(s))$; **continue**
  for all successors $s' \in \mathsf{Post}(s)$ **do** ▷ delay and edge
    $\mathrm{LB}(s') \leftarrow \mathrm{LB}(s) + \underline{c}_{\mathrm{step}}(s \to s')$
    if $\exists \hat{s} \in \mathcal{F}$ with $\hat{s} \preceq s'$ **then continue**
    remove from $\mathcal{F}$ any t with $s' \preceq t$; insert s' into $\mathcal{F}$
    push s' into Q with key $f_\omega(s') = \mathrm{LB}(s') + \omega\, h(s')$
return $UB^\star$ ▷ exact optimum if termination holds; otherwise anytime best

6. Correctness

We collect the guarantees underlying our search: sound lower bounds, safe pruning, A* optimality with admissible/consistent heuristics, and a finiteness condition for termination. Throughout, assume nonnegative rates and edge costs.

Lemma 7 (Monotone lower bounds)**.** *Along any explored path, the propagated* LB *never exceeds the true accumulated optimal cost to the current node.*

Proof. Edges add exact discrete costs; delay steps add a lower bound on required time multiplied by a nonnegative rate. Induction over steps yields the claim. □

Lemma 8 (Safe antichain pruning)**.** *If $s' \preceq s$ with $\ell(s') = \ell(s)$, then discarding s does not remove all witnesses to any solution strictly cheaper than those still achievable from s'.*

Proof. $Z \subseteq \alpha(Z')$ plus post-compatibility of α (§4) ensures simulation at the symbolic level; $\mathrm{LB}' \leq \mathrm{LB}$ prevents losing a path with a strictly better prefix bound. □

Theorem 9 (Soundness)**.** *If Algorithm 2 terminates and returns C, then C equals the optimal reachability cost from ℓ_0 to T.*

Proof. By Lemma 7 and admissibility (Lemma 5), the queue minimum $f = \mathrm{LB} + h$ is a global lower bound on any yet-unseen solution. When a target is popped or the queue is exhausted, the incumbent equals this lower bound; hence optimality holds. Lemma 8 ensures pruning does not remove all optimal witnesses. □

Theorem 10 (Conditional Termination)**.** *If, for each location ℓ, the set $\{\alpha(Z) \mid Z \subseteq \mathrm{Inv}(\ell)\}$ has finitely many minimal elements under inclusion (e.g., diagonal-aware LU bounds stabilize within finite ranges), then the frontier antichain is finite and Algorithm 2 terminates.*

Proof. Under finiteness, insertions strictly progress in a well-quasi-ordered domain (finite minimal elements per location). Combined with nonnegative costs (which forbid infinite strictly improving cycles), the priority-driven enumeration cannot create infinitely many incomparable minima. □

Corollary 11 (Weighted A*). *With $f_\omega = \mathrm{LB} + \omega h$ and $\omega \geq 1$, the first goal popped has cost at most ω times optimal.*

Proof. Standard Weighted-A* analysis applies because LB is a true path cost lower bound and h is admissible; see, e.g., the classical proof adapted by replacing exact path cost g with our certified LB. □

7. Implementation Notes

We implement zones as DBMs over *clock differences* and keep them in canonical (closed) form after every mutating operation (delay, guard intersection, reset, abstraction). Below we detail the main engineering choices.

Let $n = |X|+1$ with the dummy clock x_0. A zone is a matrix $M \in (\mathbb{Z} \cup \{+\infty\})^{n \times n}$ whose entry $M[i,j]$ encodes $x_i - x_j \leq c$ (with a strictness bit). We store: (i) a packed integer bound $(c \ll 1) \,|\, \text{strict}$, (ii) a dense $n \times n$ array for cache locality, and (iii) a small bitmap for "$+\infty$". After each operation we *close* by Floyd–Warshall; in practice, we use an *incremental closure*:

- Guard tighten: only the rows/columns touched by tightened entries are re-relaxed; average $O(n^2)$.
- Reset: updating row/column pairs for R and re-closing from those indices; average $O(|R|\, n^2)$.
- Delay (up-closure): we drop all upper bounds $x_i - x_0 \leq c$ that are not implied by invariants; this is linear in the number of finite entries, followed by a cheap re-closure.

Strictness is propagated by standard min-plus arithmetic on pairs (c, strict).

We precompute static L_{xy}, U_{xy} by a single scan of guards/invariants and store them in a dense table aligned to the DBM indices. Dynamic learning occurs during exploration: if a new constant c is observed for $x - y \backsim c$, we non-decreasingly update L, U and *invalidate* the hash cache for the affected location. Applying α to Z is a per-entry *clip* followed by incremental closure. We keep $\alpha(Z)$ alongside Z to avoid recomputation at dominance checks.

The delay and edge posts (§3) are implemented directly on DBMs. For the minimal-delay computation used in $\underline{c}^{\mathrm{delay}}_{\mathrm{step}}$ and in h_{rate}, we extract per-clock "slack to upper bound" from M and ignore diagonal literals (time-invariant) while respecting the invariant's simple bounds.

The frontier $\mathcal{F}$ is a per-location bucketed structure:

1. Key: a 64-bit *fingerprint* of $\alpha(Z)$ (Murmur-style hash on the closed DBM bytes plus the strictness bitmap).
2. Bucket: a small vector of candidates sharing the fingerprint. Exact dominance uses componentwise DBM comparison; cost-comparison is a scalar test on LB.

On insertion we run the two-phase procedure from Algorithm 1. We measure effective complexity as $O(b\, n^2)$ per candidate, with tiny b (typ., 1–5).

For each abstracted zone key we cache a bitmask of edges e with nonempty $Z \cap g_e$. Since guards rarely change structure across nearby zones, this avoids re-checking clearly-disabled transitions. We invalidate the cache on any L, U update.

We precompute $d_{\text{disc}}(\ell)$ by a multi-source Dijkstra from all targets using edge weights $w^\star$. For h_{rate}, we compute per-location $\min \text{Rate}(\ell)$ (trivial here as rates are location-constant) and then query zone slack online in $O(n^2)$.

All constants are 32-bit signed integers; we clamp to a sentinel $+\infty = 2^{31} - 1$ and guard against overflow when summing along Floyd–Warshall. Costs and LB, UB are 64-bit integers or fixed-point (Q16.16) if fractional rates are enabled (not used in our default experiments). All queues/containers are stable and deterministic under ties.

We keep the core single-threaded for determinism but allow (optional) *speculative* successor generation in a thread pool that returns candidate nodes to the main thread for antichain checks and queue insertion.

The solver stops on: (i) empty queue (exact optimum), (ii) gap $\leq \varepsilon$ (tolerance), or (iii) time/memory budget. We always report $UB^\star$ and the *certificate* $\min_{s \in Q}(\text{LB} + h)$ to quantify residual uncertainty.

8. Experiments

We evaluate three questions: (Q1) Does diagonal-aware simulation reduce explored states and time vs. diagonal-free conversion? (Q2) How much do antichains and heuristics contribute? (Q3) What is the anytime behavior under time limits?

8.1. Benchmarks

We use classical real-time models enriched with diagonal offsets:

- Fischer's mutex (offset variants). $N \in \{4, 6, 8\}$ processes with skew and mutual exclusion guards using $x_i - x_j \leq c$.
- CSMA/CD with offsets. Backoff and collision windows with inter-station offsets.
- TrainGate headway. Safety ensures $x_i - x_{i-1} \geq \Delta$ on a single-track segment.
- Job-Shop (inter-task). Release/precedence constraints as $x_{\text{start}} - x_{\text{end}} \geq c$.
- Synthetic stressors. Random TA with controlled *diagonal density* $\rho \in [0, 1]$ (ratio of diagonal to simple literals) and clock count $n \in \{6, 8, 10, 12\}$.

Each model is made *priced* by assigning nonnegative location rates and edge costs (e.g., per-unit waiting cost and switching penalties).

8.2. Baselines

1. DiagFree: diagonal-free conversion by clock-copying and additional locations, solved by a standard priced-zone engine.
2. Split: keep diagonals but apply *zone-splitting* on diagonal guards.
3. Ablations: (a) *No-AC*: our method without antichain pruning; (b) *No-H*: our method with h_0 only; (c) *No-AC+No-H*.

8.3. Protocol

We run each instance for three seeds and report medians (and IQR). The default heuristic is $h_{\oplus}$ and weight $\omega = 1.0$ (exact A*). Time limit: 600 s; memory limit: 8 GB; tolerance $\varepsilon = 0$ unless stated. For anytime plots we also test $\omega \in \{1.2, 1.5\}$.

8.4. Metrics

Wall-clock time, explored nodes, peak resident memory, optimal cost (or best $UB^{\star}$), and gap@t = $UB^{\star} - \min_{s \in Q}(\mathrm{LB} + h)$ at fixed horizons $t \in \{10, 30, 60\}$s.

8.5. Expected outcomes

(Q1) Our method should reduce node counts and time by avoiding diagonal blow-ups. (Q2) Antichain pruning cuts 30–80% of nodes; $h_{\oplus}$ improves time notably on dense-guard models. (Q3) Weighted A* achieves small gaps rapidly and tightens over time.

Table 1. Illustrative results schema (fill with your runs).

Benchmark	Nodes	Time (s)	Opt. Cost	Gap@30s	Peak Mem (MB)
Fischer-6	12300	1.6	12	0.0	72
JobShop-8	128000	62.3	305	8.0	410
TrainGate	27900	5.2	9	0.0	98

8.6. Benchmark characteristics (for context)

Include a small table with clocks, edges, diagonal density, and rate ranges:

Table 2. Benchmark characteristics (example schema).

Benchmark	#Clocks	#Edges	Diag. density ρ	Rate range
Fischer-6	7	42	0.45	[0,2]
JobShop-8	9	58	0.60	[0,3]
TrainGate	6	24	0.50	[0,1]

8.7. Ablation template

Report the incremental effect of antichains and heuristics:

Table 3. Ablations on `Fischer-6` (example schema).

Variant	Nodes	Time (s)	Opt. Cost	Gap@60s
No-AC+No-H	51800	7.4	12	0.0
No-AC	33200	3.1	12	0.0
No-H	18500	2.2	12	0.0
Ours (full)	12300	1.6	12	0.0

8.8. Reproducibility artifacts

Provide: (i) model files (`.ta` or JSON) with a README describing costs, (ii) a `run.sh` that reproduces all tables, (iii) CSVs with node/time/memory, (iv) seeded PRNG for synthetic models, and (v) a container recipe (e.g., Dockerfile) pinning compiler and library versions. We recommend emitting both raw logs and `.csv` summaries for each run.

This setup aligns with the goals from the introduction: it stresses diagonal-heavy scenarios, isolates the effects of antichains and heuristics, and quantifies anytime behavior while keeping the implementation deterministic and auditable.

9. Conclusion

We presented a cost-optimal reachability method for PTA with diagonals based on simulation, antichains, and admissible heuristics. Results indicate significant reductions in explored states while preserving exact optimality. Future work includes multi-objective costs and probabilistic extensions.

References

[1] Gastin, P., Mukherjee, S., & Srivathsan, B. (2018). Reachability in timed automata with diagonal constraints. arXiv e-prints, arXiv-1806.

[2] Larsen, K. G., & Rasmussen, J. I. (2008). Optimal reachability for multi-priced timed automata. *Theoretical Computer Science, 390*(2-3), 197-213.

[3] Larsen, K., Behrmann, G., Brinksma, E., Fehnker, A., Hune, T., Pettersson, P., & Romijn, J. (2001, July). As cheap as possible: effcient cost-optimal reachability for priced timed automata. In *International Conference on Computer Aided Verification* (pp. 493-505). Berlin, Heidelberg: Springer Berlin Heidelberg.

[4] Bouyer, P., Brihaye, T., Bruyere, V., & Raskin, J. F. (2007). On the optimal reachability problem of weighted timed automata. *Formal Methods in System Design, 31*(2), 135-175.

[5] Larsen, K. G., & Rasmussen, J. I. (2005, April). Optimal conditional reachability for multi-priced timed automata. In *International Conference on Foundations of Software Science and Computation Structures* (pp. 234-249). Berlin, Heidelberg: Springer Berlin Heidelberg.

[6] Bouyer, P., Colange, M., & Markey, N. (2016, July). Symbolic optimal reachability in weighted timed automata. In *International Conference on Computer Aided Verification* (pp. 513-530). Cham: Springer International Publishing.

[7] Boucheneb, H., Lime, D., Parquier, B., Roux, O. H., & Seidner, C. (2017, August). Optimal reachability in cost time Petri nets. In *International Conference on Formal Modeling and Analysis of Timed Systems* (pp. 58-73). Cham: Springer International Publishing.

[8] Gastin, P., Mukherjee, S., & Srivathsan, B. (2019, July). Fast algorithms for handling diagonal constraints in timed automata. In *International Conference on Computer Aided Verification* (pp. 41-59). Cham: Springer International Publishing.

[9] Alur, R., Bernadsky, M., & Madhusudan, P. (2004, July). Optimal reachability for weighted timed games. In *International Colloquium on Automata, Languages, and Programming* (pp. 122-133). Berlin, Heidelberg: Springer Berlin Heidelberg.

[10] Alur, R., La Torre, S., & Pappas, G. J. (2004). Optimal paths in weighted timed automata. *Theoretical Computer Science, 318*(3), 297-322.

[11] Fränzle, M., & Swaminathan, M. (2009, September). Revisiting decidability and optimum reachability for multi-priced timed automata. In *International Conference on Formal Modeling and Analysis of Timed Systems* (pp. 149-163). Berlin, Heidelberg: Springer Berlin Heidelberg.

[12] Panek, S., Stursberg, O., & Engell, S. (2008). Scheduling Based on Reachability Analysis of Timed Automata. *Logistic Optimization of Chemical Production Processes*, 215-235.

[13] Parrot, R., & Lime, D. (2020, August). Backward Symbolic Optimal Reachability in Weighted Timed Automata. In *International Conference on Formal Modeling and Analysis of Timed Systems* (pp. 41-57). Cham: Springer International Publishing.

[14] Woźna-Szcześniak, B., & Zbrzezny, A. (2010). SAT-based searching for k-quasi-optimal runs in weighted timed automata. Scientific Issues of Jan Długosz University in Częstochowa. *Mathematics, 15*.

[15] Kwiatkowska, M., Norman, G., Parker, D., & Sproston, J. (2006). Performance analysis of probabilistic timed automata using digital clocks. *Formal Methods in System Design, 29*(1), 33-78.

[16] Bouyer, P., Fahrenberg, U., Larsen, K. G., & Markey, N. (2011). Quantitative analysis of real-time systems using priced timed automata. *Communications of the ACM, 54*(9), 78-87.

[17] Subbiah, S., Panek, S., Engell, S., & Stursberg, O. (2007, May). Scheduling of multi-product batch plants using reachability analysis of timed automata models. In *International Conference on Informatics in Control, Automation and Robotics* (Vol. 2, pp. 141-148). SCITEPRESS.

[18] Kwiatkowska, M., Norman, G., Parker, D., & Sproston, J. (2003, September). Performance analysis of probabilistic timed automata using digital clocks. In *International Conference on Formal Modeling and Analysis of Timed Systems* (pp. 105-120). Berlin, Heidelberg: Springer Berlin Heidelberg.

[19] Larsen, K. G., Fahrenberg, U., & Legay, A. (2017). From Timed Automata to Stochastic Hybrid Games Model Checking, Synthesis, Performance Analysis and Machine Learning. In *Dependable Software Systems Engineering* (pp. 60-103). IOS Press.

[20] Srba, J. (2008, September). Comparing the expressiveness of timed automata and timed extensions of Petri nets. In *International Conference on Formal Modeling and Analysis of Timed Systems* (pp. 15-32). Berlin, Heidelberg: Springer Berlin Heidelberg.

[21] Fahrenberg, U., Larsen, K. G., & Legay, A. (2013). Model-based verification, optimization, synthesis and performance evaluation of real-time systems. In Unifying Theories of Programming and Formal Engineering Methods: International Training School on Software Engineering, Held at ICTAC 2013, Shanghai, China, August 26-30, 2013, Advanced Lectures (pp. 67-108). Berlin, Heidelberg: Springer Berlin Heidelberg.

[22] Kwiatkowska, M., Norman, G., Parker, D., & Sproston, J. Model Checking for Probabilistic Timed Automata with Digital Clocks. *Actes de l'Ecole d'Eté Temps Réel* 2005-ETR'2005, 105.

[23] Bisgaard, M., Gerhardt, D., Hermanns, H., Krčál, J., Nies, G., & Stenger, M. (2019). Battery-aware scheduling in low orbit: the GomX–3 case. *Formal Aspects of Computing, 31*(2), 261-285.

[24] Waez, M. T. B., Dingel, J., & Rudie, K. (2011). Timed automata for the development of real-time systems. *Research Report 2011*, 579.

[25] Yagoubi, R. S., Naud, O., Dejean, K. G., & Crestani, D. (2018). New approach for differential harvest problem: The model checking way. *IFAC-PapersOnLine, 51*(7), 57-63.

How to cite this article: Sayan Mukherjee (2020). Cost-Optimal Reachability for Timed Automata with Diagonals via Zone Simulations. *Bulletin of Computer and Data Sciences*, 1(1), 3-18. DOI: 10.71448/bcds2011-2

Received: 30/5/2020 **Revised:** 20/6/2020 **Accepted:** 25/9/2020 **Publish:** 30/12/2020