Lazy and Priority-Guided Product Construction for Non-Integer Discounted-Sum Synthesis

Raj Chandak and Priyanka Golia

Department of Computer Science and Engineering, IIT Delhi
{cs5221643,pgolia}@cse.iitd.ac.in

Abstract. Reactive synthesis with non-integer discounted-sum objectives is solved by Bansal et al. (AAAI 2022) via the DSLow comparator automaton, which under-approximates the running sum and reduces the problem to a Büchi game on the product of the arena and comparator. Their construction allocates the full product upfront and explores it by BFS, incurring two costs: many product states are structurally unreachable (*ghost states*), and BFS defers winning states behind a wide shallow frontier. We propose (1) a *lazy* on-the-fly construction that materializes only reachable states, and (2) a *priority-guided* exploration keyed on $U - c$ (distance to comparator saturation) with interleaved backward propagation. On the standard benchmarks, our approach constructs 74–99% fewer states and discovers the first winning state up to 22× faster, with identical winning regions. Weight-specific asymmetric comparator bounds additionally cut the theoretical state space by 40–50% at no cost.

Keywords: Reactive synthesis · Discounted-sum games · Comparator automata

1 Introduction

The *discounted sum* $\sum_{i\geq 0} w(e_i)/d^i$ with discount $d > 1$ is a standard quantitative objective in reactive synthesis. For integer d the problem reduces to an ω-regular game, but for *non-integer* d (e.g., d=1.5) the running sum takes values in an unbounded set. Bansal et al. [3] resolve this with the DSLow comparator automaton: a finite integer *under-approximation* of the running sum that soundly detects when the threshold is met. This reduces synthesis to a Büchi game on the *product* $\mathcal{P}$ of the arena (vertex set V, initial v_0) and the comparator, where a state (v, c) is winning iff v satisfies the LTL goal and c equals the upper comparator bound U. Comparator automata originate in discounted-sum inclusion checking [1,2]; related on-the-fly and heuristic-guided constructions appear for LTL synthesis (Strix [5]), LTLf synthesis [6], and LTLf planning [7]; the multi-agent satisficing setting is studied in [4].

Two inefficiencies limit the DSLow baseline. *(P1) Ghost states:* the comparator update is non-surjective from $(v_0, 0)$, so many values in the comparator range $[L, U]$ are structurally unreachable, yet the baseline allocates all $|V| \cdot (U{-}L{+}1)$ pairs. *(P2) No progress signal:* BFS builds a wide shallow frontier before any winning state (v, U) is reached, so backward propagation contributes nothing early. Concretely, on Social Distancing 10×10, B0's symmetric bounds give a theoretical product of ≈13.3M pairs; even after asymmetric tightening (B1) this shrinks to ≈7.3M, of which only 1.1% are reachable. BFS still builds 9,332 states (12% of reachable) before any propagation. Our lazy construction avoids ghost states; our priority-guided construction with interleaved propagation often terminates before the full reachable product is built. Together they reduce states built by 74–99% and find the first winning state up to 22× sooner, with no change to the winning region.

Preliminaries. We fix $d = 1 + 2^{-k}$ for integer $k \geq 1$, with precision parameter p controlling the approximation error $\leq 2^{-(p+k)}$. Write $R = 2^{k+p}$, $D = 2^k$, $R_t = 2^{2k+p}$. A DSLow comparator state $c \in [L, U]$ advances a scaled weight with a rounding correction, bounded to $[L, U]$:

$$c' = \operatorname{clamp}\big(c + wR + \lfloor c/D \rfloor,\ L,\ U\big).$$

The arena G has vertex set V (which already includes the LTL specification automaton) and initial vertex v_0; the product $\mathcal{P}$ pairs arena vertices with comparator values and is solved by backward induction from winning states $\{(v, U) \mid v \text{ LTL-accepting}\}$. When edge weights are asymmetric ($w^+ \neq w^-$), the range tightens from the symmetric $W = \max_e |w(e)|$ default to $L = w^- \cdot R_t$, $U = w^+ \cdot R_t + D$, shrinking the comparator by 40–50%; we apply this as preprocessing (configuration B1 below).

2 Optimizations

Lazy construction (P1). We start from $(v_0, 0)$ and create (v', c') only on first visit, using a hash map and a visited set; ghost states are never materialized. Savings scale with $1 -$ fill rate, where fill $= |\text{reachable}|/|V| \cdot (U - L + 1)$. Table 1 shows the effect is large: Grid World reaches only 0.74% of its theoretical states (the baseline would process ≈135× more states than any exploration could visit), and even the densest case (Conveyor) has only 26.4% fill.

Table 1. Fill rates under B2 (lazy). sd $n \times n$: rewards +10/−2, k=2, p=1. GW/Conv: k=1, p=1. $|V|$ includes LTL automaton.

Benchmark	$\|V\|$	Theoretical	Reachable	Fill
sd 4×4	397	141,729	14,568	10.3%
sd 6×6	2,407	859,299	34,574	4.0%
sd 8×8	8,093	2,889,201	51,056	1.8%
sd 10×10	20,572	7,344,204	77,576	1.1%
Grid World (GW)	20,572	2,694,932	20,003	**0.74%**
Conveyor (Conv)	14,133	494,655	130,713	26.4%

Table 2. States to first win ("fw"): BFS (B2) vs. priority queue (B3). Improv = B2 fw / B3 fw.

Benchmark	Total	B2 fw	B3 fw	Improv
sd 4×4	14,568	976	45	**21.7×**
sd 6×6	34,574	2,730	205	**13.3×**
sd 8×8	51,056	5,377	459	**11.7×**
sd 10×10	77,576	9,332	546	**17.1×**
GW	20,003	4,136	505	**8.2×**
Conv	130,713	88	52	1.7×

Priority-guided construction (P2). $U - c$ measures the comparator's distance from becoming winning: a state's hop to a winning label is exactly how much more c must grow before hitting U. This is not benchmark-specific—it follows directly from the DSLow winning condition. Replacing the FIFO queue with a min-heap keyed on $U - c$ expands the nearest-to-saturation state first (greedy best-first, not A*). Each time a winning state is created, we propagate immediately through already-built reverse edges; if this resolves $(v_0, 0)$, we terminate (*construction early termination*). This parallels Strix [5]; $U - c$ is the discounted-sum analog of Strix's heuristic.

Correctness. Priority ordering changes *when* each state is visited, not *whether*; lazy BFS and priority-queue exploration visit the same reachable states and derive the same labeling under backward induction. Early termination is sound because labeling is monotone—adding states or edges can only propagate *more* winner labels, never retract existing ones, so once $(v_0, 0)$ is labeled winning it remains so.

3 Experiments

Setup. Three domains from [3]: Social Distancing ($n \times n$ grid, rewards $+10/-2$, $n \in \{4, 6, 8, 10\}$); Grid World (asymmetric banana collection); Conveyor Belt (symmetric weights $\{-2, \ldots, 2\}$). Runs use $k \in \{1, 2\}$ and $p = 1$. Configurations compose: B0 = baseline BFS [3]; B1 = B0 + asymmetric bounds; B2 = B1 + lazy; B3 = B2 + priority queue with interleaved propagation. Implementation is C++ on NonIntegerGames[1], single-threaded. Every configuration returns the same winning region as B0 across all 12 instances. We report states constructed (rather than wall-clock time) to isolate the algorithmic effect.

RQ1: does lazy construction eliminate ghost states? Yes, and how much it helps is determined by geometry, not comparator size. Table 1 shows fill rates ranging from 0.74% (Grid World) to 26.4% (Conveyor); B2 avoids 74–99% of the states B1 allocates. The floor-and-clamp in the comparator update is non-surjective, concentrating reachable states in a narrow band of $[L, U]$.

RQ2: does priority-guided exploration reduce states-before-first-win? Yes, substantially—but only when weights are asymmetric. Table 2 reports 8.2–21.7× reductions on the five asymmetric benchmarks. On Conveyor Belt, whose nearly-symmetric weights $\{-2, \ldots, 2\}$ give $U - c$ no directional gradient, the improvement collapses to 1.7×—confirming the mechanism: no preferred direction means no gradient for $U - c$.

Combined effect. On a representative asymmetric instance ($\mathrm{sd}_{8\times 8}$), the three optimizations attack complementary costs: asymmetric bounds contract the allocated theoretical space by 1.8×, lazy construction then skips ghost states for a further 56.6× reduction (from B1 theoretical to the 51,056 actually reachable states), and the priority queue finds the first winning state 11.7× sooner, triggering early termination before BFS would surface any winning state. Conveyor Belt is the principled exception, as in RQ2.

[1] `https://github.com/aarcee17/NonIntegerGames`

References

1. S. Bansal, S. Chaudhuri, M. Y. Vardi. Comparator automata in quantitative verification. FoSSaCS 2018.
2. S. Bansal, M. Y. Vardi. Safety and co-safety comparator automata for discounted-sum inclusion. CAV 2019.
3. S. Bansal, L. E. Kavraki, M. Y. Vardi, A. Wells. Synthesis from satisficing and temporal goals. AAAI 2022, pp. 9679–9686.
4. S. Rajasekaran, S. Bansal, M. Y. Vardi. Multi-agent systems with quantitative satisficing goals. IJCAI 2023, pp. 280–288.
5. P. J. Meyer, S. Sickert, M. Luttenberger. Strix: Explicit reactive synthesis strikes back! CAV 2018.
6. S. Xiao, J. Li, S. Zhu, Y. Shi, G. Pu, M. Y. Vardi. On-the-fly synthesis for LTL over finite traces. AAAI 2021, pp. 6530–6537.
7. Y. Suprun, K. Elimelech, L. E. Kavraki, M. Y. Vardi. TIDE: A trace-informed depth-first exploration for planning with temporally extended goals. arXiv:2601.12141, 2026.