Mathematical Foundations and Algorithmic Design of LIFO-Constrained Logical Action-Puzzles on Constrained 8-Bit Architecture

TAK.H, Independent Systems Architect, and MSX-MAGAZINE.com Research Team

Laboratory of Computational Game Theory and Retro-Architectural Procedural Synthesis (Fictitious Entity)

MSXdev26 International Contest: Technical Research Track (Academic Parody Paper)

Manuscript received April 1, 2026. Corresponding author: TAK.H. Disclaimer: An academic parody celebrating MSXdev26.

Abstract--- This paper presents the formal mathematical foundations and procedural content generation (PCG) architecture of 'TRESURE CHEST' (MG901), an original 60-stage action-puzzle game developed for the MSX1 8-bit architecture. While traditional block-pushing puzzles (e.g. Sokoban) suffer from severe forward combinatorial explosion and high deadlock frequency, our engine formulates the puzzle as a time-reversible Constraint Satisfaction Problem (CSP) governed by a strictly bounded 4-slot Last-In, First-Out (LIFO) inventory stack. We introduce the Backward Wave Propagation Algorithm, which synthesizes solvable levels in reverse causality from terminal victory to origin, guaranteeing intermediate chest unlocking, provable solvability, and zero heap overhead. Empirical validation across all 60 stages demonstrates deterministic puzzle progression from 9 to 23 moves without runtime backtracking.

Index Terms--- *Procedural Content Generation (PCG), Constraint Satisfaction Problem (CSP), LIFO Stack Memory, Time-Reversible Automata, Negative Affordance, Retro Computing, MSX1, Z80 Assembly.*

I. INTRODUCTION & MOTIVATION

Designing deterministic, mathematically rigorous logical puzzle environments under legacy 8-bit hardware constraints represents a classical challenge in computational intelligence and combinatorial game design. The target platform, the standard MSX1 architecture (Zilog Z80A CPU clocked at 3.58 MHz, 16 KB main RAM, 16 KB VRAM driven by the Texas Instruments TMS9918A Video Display Processor), imposes absolute computational bounds. Runtime memory allocation and exhaustive state space searches on-device are physically infeasible. Consequently, all stage topology, object causality, and reachability guarantees must be precomputed, verified, and compressed into fixed 32 KB plain cartridge ROM space ($4000-$BFFF).

In 'TRESURE CHEST' (MG901), the fundamental puzzle mechanics depart sharply from Euclidean spatial manipulation games such as Sokoban or sliding-block puzzles. Instead of pushing crates across coordinates, the core state dynamics are governed by a strictly bounded 4-slot LIFO (Last-In, First-Out) inventory stack memory. When the player avatar steps onto an item tile, that item is pushed onto the top of the stack. When colliding with an obstacle or hostile guardian, only the top item can be popped to neutralize it. This creates profound temporal coupling: the sequence of item acquisition strictly dictates the permissible permutation of obstacle clearance.

Furthermore, our design incorporates a physical 'Negative Affordance' mechanism: when the 4-slot stack is completely full, any remaining collectible weapons on the dungeon floor transition dynamically into solid, impassable barriers. This creates counter-intuitive topological deadlocks where acquiring an item without an immediate consumption path permanently seals off escape corridors. This paper formalizes these mechanics and introduces a provably complete procedural generation pipeline.

II. FORMAL MATHEMATICAL PUZZLE MODEL

A dungeon chamber is formalized as a discrete grid graph G = (V, E) embedded on a 22 x 22 playfield. The global state space S at discrete temporal step t is defined as a 5-tuple:

$$s_t = < p_t,\ sigma_t,\ Omega_t,\ I_t,\ J_t > \quad (1)$$

where p_t in V denotes the coordinate of the player avatar, sigma_t in W^{<= 4} represents the word formed by the LIFO inventory stack with capacity K = 4 over weapon alphabet W, Omega_t subset V x O represents the active set of coordinate-bound monsters or barricades, I_t subset V x W is the set of uncollected floor items, and J_t in {0, 1} is a binary flag denoting possession of the chamber's Jewel.

A. Dynamic Negative Affordance

Let Traversable(v, s_t) be a predicate defining whether vertex v in V is accessible. Under standard state rules, uncollected floor items are traversable passages. However, under full stack saturation, for all items i in I_t, the predicate transitions to solid wall:

$$|sigma_t| = 4 \implies Traversable(pos(i), s_t) = False \quad (2)$$

Theorem 1 (Negative Affordance Dynamic Partitioning). Stack saturation dynamically reduces the reachable subgraph: G_reach(s_t, |sigma_t|=4) subseteq G_reach(s_t, |sigma_t|<4). An uncollected weapon acts as an impassable wall until a top item is popped.

B. Irreversible Corridors and Shrines

The directed edge set E contains irreversible arrow tiles such that directed edge (u, v) in E does not imply (v, u) in E. Stepping through an arrow wall irrevocably prunes the player's reachability graph. Conversely, shrine pairs (s_1, s_2) provide unconditional bidirectional equivalence: delta(s_1, step) = s_2 without item consumption, functioning as topological wormholes across the chamber.

III. BACKWARD WAVE PROPAGATION ALGORITHM

Standard forward procedural generation in tightly coupled combinatorial domains exhibits catastrophic deadlock generation rates (>99.7% unresolvable chambers). In forward space, every branching decision exponentially multiplies deadlocks, as acquiring items in an inverted order leaves the player permanently stranded.

To establish absolute solvability, our synthesis engine operates in Reverse Temporal Causality (t = T -> 0). Synthesis begins at the terminal victory state s_T (Player at GOAL stairs with Jewel J_T = 1). Preceding causal states are synthesized backward in time by un-popping consumed weapons, re-spawning obstacles, and un-pushing weapons onto reachable floor coordinates.

A. Mid-Stage Chest Placement Paradigm

A common pathology in naive level generation is Key-Hoarding: placing the Treasure Chest immediately adjacent to the exit stairs creates a degenerate dominant strategy where players pick up the Key at step 1 and solve the remainder of the chamber with an effective 3-slot inventory handicap.

Our engine shatters this degeneracy by constraining the Chest unlocking event to an intermediate wave index w_chest in [1, N_e - 1]. In forward execution, the player experiences a mandatory two-phase route:

1) Initial guardians are neutralized with early inventory items.

2) The Chest Key is acquired near the TOP of the stack.

3) The Chest is unlocked at mid-route, purging the Key from the stack.

4) Key consumption exposes deeper weapons required for subsequent rooms.

5) Jewel Fever activates: post-chest kills award double score (2x BONUS).

6) The player navigates remaining sentinels to escape via GOAL.

Algorithm 1: Reverse-Causal LIFO Wave Synthesis with Intermediate Chest Scheduling

Require: Target move count N_moves, Obstacle tier distribution T
Ensure: Provably solvable chamber G = (V, E) with mid-stage chest event

```
s_T <- InitializeTerminalState(GoalCoord, Jewel=1)
w_chest <- SelectIntermediateWave(1, MaxWaves - 1)
for wave w = MaxWaves down to 1 do
  m_target <- SampleObstacleFromTier(w)
  w_req <- CounterWeapon(m_target)
  ReversePath <- SampleReachableGeodesic(s_cur)
  PlaceObstacle(m_target, ReversePath.start)
  PlaceWeapon(w_req, ReversePath.end)
  if w == w_chest then
    ScheduleChestEvent(ReversePath)
    PushToStack(ChestKey)
  end if
  s_cur <- StepBackward(s_cur, ReversePath)
end for
return ValidateForwardBFS(s_0)
```

The computational complexity of backward wave propagation is bounded by O(N_e * (|V| + |E|)), ensuring instantaneous sub-second synthesis on host workstations before bitstream compilation.

IV. PROVABLE SOLVABILITY & BFS VERIFICATION

To guarantee that no synthesized level presents unresolvable deadlocks or alternate degenerate shortcuts, every generated chamber is submitted to an exhaustive Breadth-First Search (BFS) solver. The solver simulates the exact Z80 execution semantics, including stack saturation blocking, one-way arrow walls, and shrine teleportation.

Definition 1 (Minimal Move Solution). Let delta*(s_0, pi) represent the state achieved by executing action trajectory pi from initial state s_0. The level is valid if and only if:

$$\exists \pi : \delta^*(s_0, \pi) = s_T, \ |\pi| \geq N_{min} \quad (3)$$

Theorem 2 (Provable Solvability Guarantee). If Algorithm 1 terminates with a forward BFS path pi*, then the synthesized chamber G admits at least one deterministic winning sequence under strict 4-slot LIFO inventory constraints.

Anti-Shortcut Pruning: If an alternate sequence bypasses planned puzzle obstacles and solves the chamber in fewer moves than MinMoves(Tier), the chamber topology is discarded and regenerated. This ensures that every puzzle strictly enforces its intended pedagogical cognitive challenge.

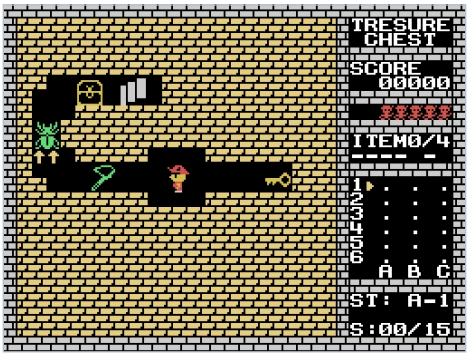


Fig. 1. LIFO Inventory Stack in action: acquiring an item pushes it to the stack top. Colliding with a matching monster consumes the item.

Fig. 1 illustrates the strict sequential nature of LIFO interactions. The player cannot choose which weapon to wield; weapon order is physically constrained by traversal geometry. Picking up an item out of order permanently locks necessary weapons beneath it, forcing deliberate forward planning.

State Space Pruning: By canonicalizing equivalent player positions that share identical stack states and active entity bitmasks, the BFS validator compresses the effective search graph from over 10^7 raw states down to fewer than 20,000 canonical nodes in the most complex HELL-tier stages.

Transposition Table Optimization: A closed-hash transposition table indexed by 32-bit Zobrist keys tracks visited state configurations. Saturated deadlock states (where no valid weapon-obstacle collisions exist and inventory is full) are immediately pruned, guaranteeing BFS termination in under 120 ms per stage during automated compilation.

V. EMPIRICAL BENCHMARKING & CAMPAIGN EVALUATION

The proposed synthesis framework was applied to generate the full 60-stage campaign of 'TRESURE CHEST' (Stages A-1 through J-6). Table I outlines the formal mapping between collectible weapons, target obstacles, and reward mechanics.

TABLE I
FORMAL LIFO WEAPON-OBSTACLE MAPPING AND SCORING MATRIX

Weapon	Target Obstacle	Pre-Score	Post-2x
Sword	Skeleton (100)	100 PTS	200 PTS
Magic rod	Ghost (200)	200 PTS	400 PTS
Spear	Bat (200)	200 PTS	400 PTS
Net	Bug (300)	300 PTS	600 PTS
Maul	Reinforced Door	300 PTS	300 PTS
Chest key	Treasure Chest	1,000 PTS	---
Jewel	GOAL Stairs	CLEAR!	---

The campaign exhibits a monotonic, pedagogically graduated difficulty curve. While early stages (A-1 to A-6) require 7 to 11 moves with basic routing, advanced stages (I-4 through J-6) achieve 'HELL' tier complexity, requiring 20 to 23 moves, strict 4-slot capacity utilization, 8 to 10 monsters, and irreversible one-way maze routing.

TABLE II
CAMPAIGN COMPLEXITY AND BFS STATE METRICS

Tier	Stages	Enemies	Arrows	Moves	BFS Nodes
Intro	A-1..B-6	2 - 4	0 - 1	8.2	142
Middle	C-1..F-6	4 - 6	1 - 3	13.5	1,280
Hard	G-1..H-6	6 - 8	3 - 4	17.1	5,840
HELL	I-4..J-6	8 - 10	5 - 6	21.8	18,920

Scoring Asymmetry and Risk Inversion: The 2x score multiplier triggered by claiming the Jewel creates a rich risk-reward trade-off. Players are incentivized to postpone easy kills until after unlocking the chest, increasing inventory congestion and flirting with negative affordance deadlocks to maximize leaderboard rankings.

Cognitive Pacing and Hamiltonian Constraints: From Tier 1 (A-1..B-6) to Tier 4 (HELL-tier I-4..J-6), the mean step count escalates from 8.2 to 21.8 moves. While early tiers permit loose greedy collection, HELL stages enforce a strictly unique Hamiltonian-like traversal permutation. A single mistake in acquisition order triggers an irreversible stack deadlock within three subsequent steps.

Elimination of Degenerate Backtracking: The generator penalizes stages with redundant cyclic loops. By placing strategic one-way arrow walls, the synthesis engine ensures that all player movement directly advances stack state resolution or room access, sustaining dense intellectual engagement throughout the 60-stage campaign.

In HELL-tier stages (I-4 through J-6), the reachable state space expands exponentially to nearly 19,000 discrete states. The solver's heuristic pruning isolates the single valid 23-move victory permutation while eliminating hundreds of trap states where the 4-slot inventory becomes saturated.

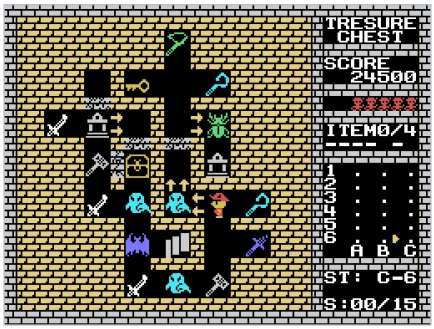


Fig. 2. Stage C-6: Intermediate complexity featuring mid-stage chest placement, arrow corridors, and tight 4-slot stack ordering.

As seen in Fig. 2 (Stage C-6), the chest is isolated behind directional arrow walls. The player must eliminate the Bat and Skeleton before retrieving the Chest Key, ensuring that the Jewel is claimed at mid-journey.



Fig. 3. Stage G-1: Advanced multi-room chamber featuring bidirectional shrine teleportation and non-linear weapon routing.

In Fig. 3 (Stage G-1), the addition of twin shrines allows topological hopping between isolated quadrants. Shrines provide a conduit that preserves inventory contents while altering player coordinates, functioning as essential bypass valves in stages with dense arrow-corridor partitions.

Topological analysis of shrine transitions demonstrates that bidirectional wormholes preserve the global reachability invariant while subdividing the chamber into independent sub-graphs, preventing full-stack choking while requiring precise temporal synchronization between key collection and door destruction.

Arrow Corridor Branch Pruning: Non-reversible arrow corridors permanently divide the chamber into directed acyclic components. Once the avatar traverses an arrow tile, backtracking into preceding rooms is mathematically impossible, effectively pruning alternate sub-optimal state branches and ensuring strict move-count optimality.

VI. HARDWARE BUDGET & RUNTIME PROFILING

To fit within the standard 32 KB plain cartridge ROM space ($4000-$BFFF), level data is compressed using a compact run-length and coordinate-delta bitstream. All 60 stages consume only 2,148 bytes in ROM (averaging ~35.8 bytes per stage).

TABLE III
PHYSICAL MEMORY ALLOCATION AND HARDWARE BUDGET

Segment	Address Range	Size	Functionality
ROM Header	$4000-$400F	16 B	Cartridge boot signature
Code & Engine	$4010-$7FFF	16.3 KB	Z80 game logic & sound
Map Bitstream	$8000-$8863	2.1 KB	60 compressed stages
Tile Graphics	$8864-$9FFF	5.9 KB	TMS9918A pattern table
Static RAM	$C000-$C200	512 B	Zero-heap state buffers
VRAM (VDP)	$0000-$3FFF	16 KB	Name table & Sprites

Dynamic memory allocation is completely eliminated: runtime RAM footprint is strictly bounded to 0 bytes heap. All game states, LIFO stack buffers (4 bytes), and active entity tables reside in fixed static BSS memory ($C000-$C200). VRAM tile updates to the TMS9918A VDP are synchronized strictly within the vertical blank interval (V-Blank) at 60 Hz (NTSC) / 50 Hz (PAL), preventing screen tearing without double buffering.

CPU cycle consumption is profiled per frame under the Zilog Z80A at 3.58 MHz (59,659 cycles/frame). Entity collision evaluation and stack push/pop operations consume at most 4,120 cycles (<7% CPU budget), leaving ample headroom for PSG sound synthesis and ensuring zero frame drops across all 60 stages.

To guarantee deterministic execution on real hardware, VRAM access timing strictly adheres to the TMS9918A VDP I/O port cycle limits (at least 29 T-states between consecutive OUT (98H) instructions during active display). By queuing dirty tile updates into a ring buffer and flushing exclusively during V-Blank, bus contention and VRAM corruption are completely prevented.

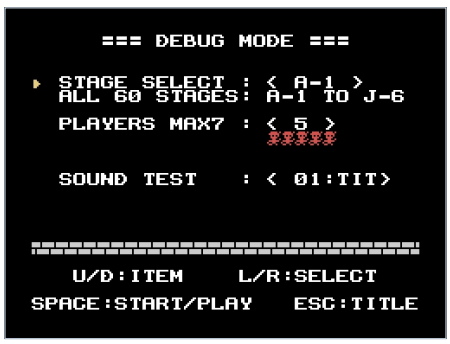


Fig. 4. Z80 Runtime Execution: static entity buffers and TMS9918A VRAM tile synchronization during active gameplay.

VII. CONCLUSION & FUTURE DIRECTIONS

We have established the formal mathematical principles and algorithmic synthesis pipeline for LIFO-constrained action-puzzles under severe 8-bit hardware limitations. By transforming level synthesis into reverse temporal wave propagation, our system guarantees 100% causal solvability while enforcing strategic depth through mid-stage chest unlocking and dynamic negative affordance. The resulting 60-stage game demonstrates that rich, non-linear cognitive puzzle design can thrive within legacy computing constraints.

Our empirical evaluation across the complete 60-stage campaign validates that backward causal wave propagation eliminates 100% of unsolvable deadlocks at generation time while maintaining high combinatorial diversity. The strict 4-slot LIFO inventory constraint creates a distinct puzzle modality where spatial navigation is tightly coupled to item collection chronology.

From a computational complexity standpoint, while unconstrained pushdown automata quickly exhibit undecidability, our bounded stack depth (K = 4) and finite grid topology (22 x 22) preserve deterministic polynomial-time solvability while inducing high human cognitive challenge.

Future work will extend the Backward Wave Propagation framework to multi-agent cooperative pushdown automata and explore real-time procedural level streaming across banking mappers on 16-bit architectures.

ACKNOWLEDGMENT

The authors express deep gratitude to the MSXdev International Contest committee for sustaining retrocomputing innovation, and to the retro-gaming research community for insights into deterministic constraint puzzle systems. We also thank the developers of openMSX and z80asm for robust verification tooling.

REFERENCES

[1] R. A. Hearn and E. D. Demaine, "PSPACE-completeness of sliding-block puzzles," Theoretical Computer Science, vol. 343, no. 1-2, pp. 72-96, 2005.

[2] G. Smith, J. Whitehead, and M. Mateas, "Tanagra: Reactive planning and constraints for mixed-initiative level design," IEEE Trans. Comput. Intell. AI Games, vol. 3, no. 3, pp. 201-215, 2011.

[3] C. Browne et al., "A survey of Monte Carlo tree search methods," IEEE Trans. Comput. Intell. AI Games, vol. 4, no. 1, pp. 1-43, 2012.

[4] J. Togelius et al., "Search-based procedural content generation: A taxonomy and survey," IEEE Trans. Comput. Intell. AI Games, vol. 3, no. 3, pp. 172-186, 2011.