Supplementary File S1: Representative Rule-Level Simulation Walkthroughs

Level 1

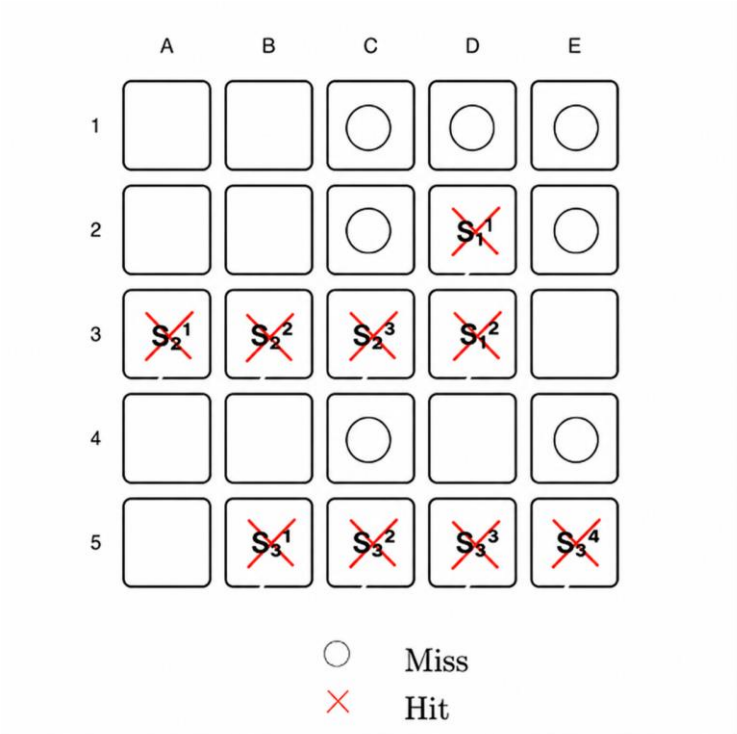

Figure S1: Level 1 target-grid example and notation for recorded outcomes. An example 5×5 target grid (A–E by 1–5) showing the shot history for a Level 1 simulation; open circles denote misses and red X marks denote hits. Hit-label notation used in the figure: symbols of the form S_j^k indicate the ship (j) and the ordered segment index (k) associated with that labelled square in the example state. These features are carried over to figures S2-S4. Summary of what the visualization provides: the figure illustrates how Level 1 outcomes are tracked spatially on the target grid and how hit events are annotated at the segment level

In this representative Level 1 run, each move is described by the recorded damage outcome and the hidden vulnerability outcome stored in the log. The game opened with a bomb-labelled shot at C3. The vulnerability outcome permitted damage, so the move was recorded as a hit. After several nearby no-damage outcomes during local search, bomb-labelled shots at B3 and A3 produced consecutive damage outcomes that sank Ship S_2 by move 6.

Later in the run, the recorded shots used torpedo labels under the shared shot routine. The targeting policy still selected coordinates rather than making a separate weapon-choice decision. Ship S_1 was destroyed by move 12. Moves 14–16 then produced three consecutive no-damage outcomes. These outcomes illustrate stochastic variation under the L1 vulnerability rule rather than a separate strategic weapon-switching effect. This sequence was less favourable than the corresponding L0 behaviour because L1 can prevent an occupied target coordinate from converting into damage when the vulnerability outcome does not permit damage. In L0, by contrast, damage depends only on whether the targeted coordinate

contains an active ship segment. Finally, a torpedo-labelled hit at move 17 restarted damage progress, and the run ended by move 20.

Level 2

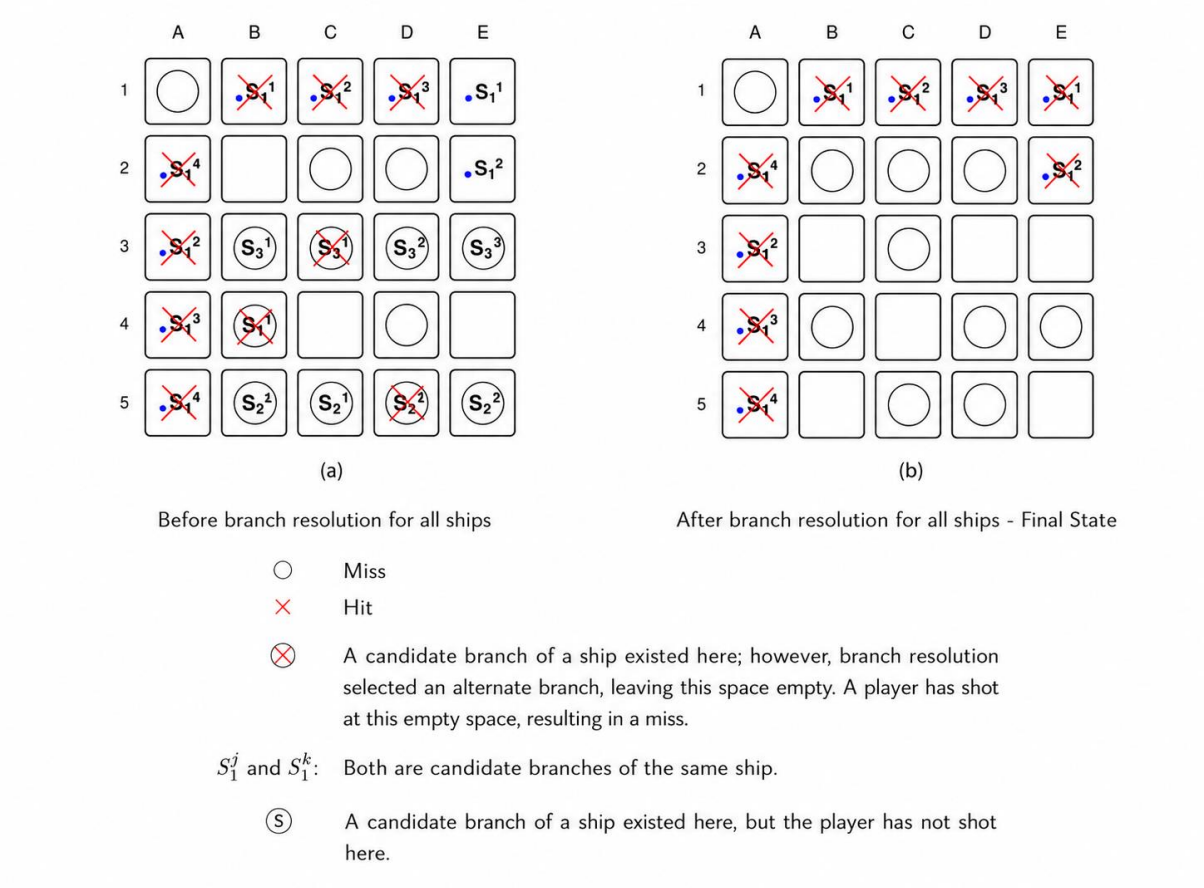

Figure S2. Level 2 branch-based placement before and after branch resolution. (a) Pre-resolution grid state showing retained candidate ship branches before the active branch is selected. (b) Post-resolution grid state after one active branch has been selected for each branch-based ship. Open circles indicate misses or no-damage outcomes, red X marks indicate recorded damage outcomes, and marked branch cells indicate coordinates where a candidate branch existed before resolution but was not selected as active. The figure shows how recorded shot outcomes are interpreted relative to the selected active branch while the final board state remains classical.

At Level 2, each ship has two retained candidate placement branches, represented using labels such as S_j^k and $*S_j^k$. In this representative run, each move is described by the recorded damage outcome and branch-resolution state stored in the log. The game opened with a bomb-labelled shot at C3. The vulnerability outcome permitted damage, but the branch-selection event selected the alternate active branch. The final recorded result was therefore no damage to the selected active placement.

This illustrates how L2 can produce a no-damage outcome even when the targeted coordinate is consistent with one retained branch, because the selected active branch may differ from the branch suggested by the shot. Compared with L0, this adds an additional hidden-state step between targeting a coordinate and applying damage. The player may therefore receive

weaker or delayed information from a move, depending on when branch resolution occurs and which branch becomes active.

After the opening sequence, the run continued through local probing around the initial area using the same targeting policy. The first ship, $*S_2$, was destroyed by move 6. After move 6, the run continued with torpedo-labelled shots under the shared shot routine, while the targeting policy continued selecting coordinates from the observed board state. The midgame included recorded bomb and torpedo labels produced by the same attack-type routine used across the simulation. These labels should not be read as a separate strategic weapon-choice policy.

The run continued until move 18, after which consecutive torpedo-labelled damage outcomes at E1 and E2 led to the destruction of the final ship, S_1. Move 15 resolved ship S_1 to branch $*S_1$. After that point, later shots were evaluated against the selected active branch and could contribute to the final sink. This walkthrough, therefore, shows how L2 adds branch-resolution uncertainty while still ending in an ordinary classical board state after the active branches have been selected.

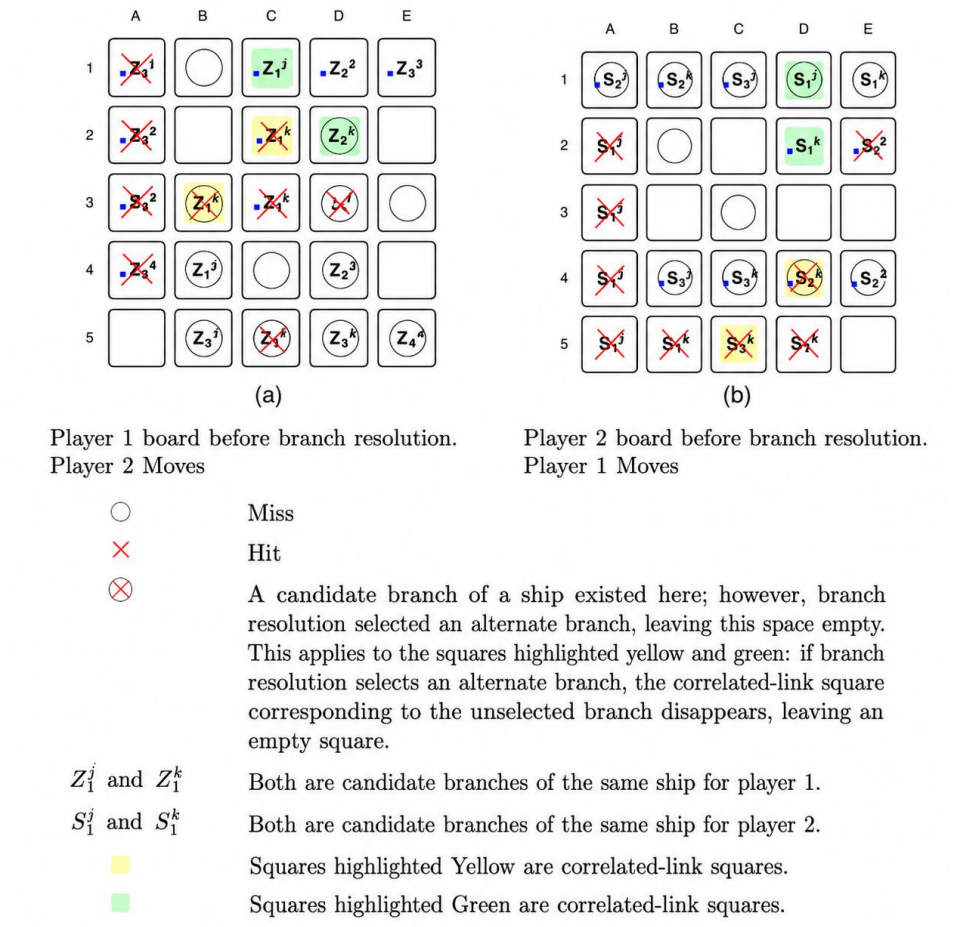

Figure S3. Level 3 player-specific boards before branch resolution.
Pre-resolution boards from a representative Level 3 run. (a) Player 1's board showing Player 2's recorded shots. (b) Player 2's board showing Player 1's recorded shots. Open circles indicate misses or no-damage outcomes, red X marks indicate recorded damage outcomes, and paired branch labels show retained candidate placements before resolution. Highlighted cells mark predefined correlated-link coordinates used by the Level 3 linked-damage rule. These markings represent classical linked coordinates, not physical quantum entanglement between ship segments.

In the representative Level 3 run, each move is described by the recorded damage outcome, hidden vulnerability state, branch-resolution state, and correlated-damage resolution. Early play consisted of probing by Player 1 and Player 2, with each shot recorded as a local outcome on the attacker's target grid. Player 1 recorded a bomb-labelled hit by move 3, and Player 2 recorded a torpedo-labelled hit by move 1.

Soon after, the first correlated-damage event occurred when Player 2 hit segment $*Z_1^1$ at C2 with a torpedo-labelled shot. The correlated-damage rule then applied an additional linked-damage update to the predefined linked coordinate on Player 2's board. This update was not available in L2. It changed the later observable board state for Player 1 because the linked-damage rule affected Player 2's board during the same Level 3 rule process.

Player 1 later finished Ship S_3 by move 10 after a sequence of bomb-labelled damage outcomes. During the middle of the run, several turns produced no damage under the layered stochastic rules. For example, Player 2's torpedo-labelled shot at D3 had a vulnerability outcome that permitted damage, but the branch-resolution outcome selected the alternate branch, so the final recorded result was no damage, and the ship resolved to the alternate branch $*Z_2$. This example shows how L3 inherits the L1 vulnerability rule and the L2 branch-resolution rule while adding a separate correlated-damage mechanism.

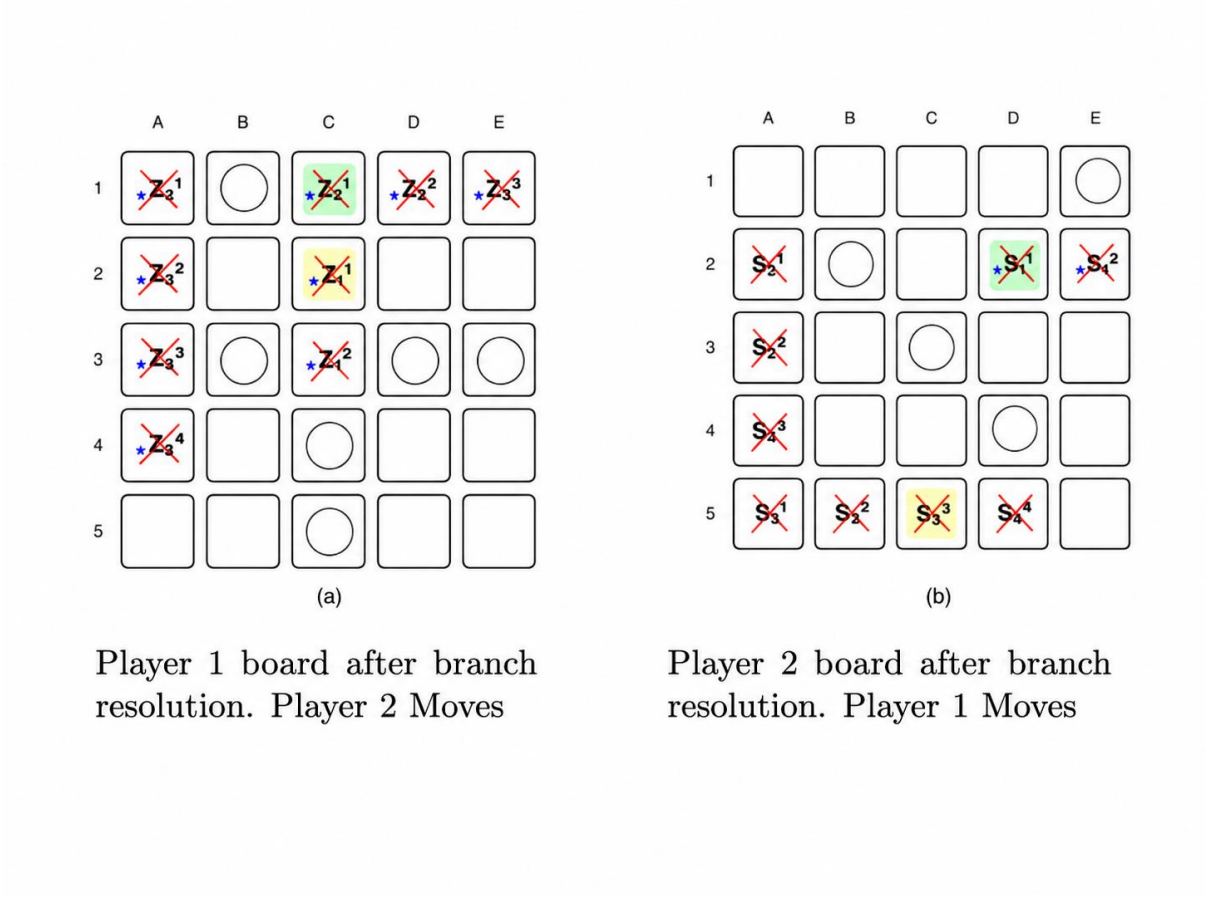

Figure S4. Level 3 player-specific boards after branch resolution.
Post-resolution boards from the same representative Level 3 run. (a) Player 1's board showing Player 2's recorded shots. (b) Player 2's board showing Player 1's recorded shots. The selected active branches define the final classical board configurations used to interpret previously recorded shot outcomes. Highlighted cells mark predefined correlated-link coordinates that could receive additional linked damage under the Level 3 rule.

Late in the run, a final correlated-damage resolution after Player 2's torpedo-labelled shot at C1 caused both fleets to become fully sunk in the same terminal update, producing a draw by move 15. This draw condition was available in L3 because linked damage could be applied during the same move as direct target damage. Under the implemented L0 rule, damage was applied only to the directly targeted board, so the recorded L0 trials did not produce simultaneous linked terminal damage. This walkthrough, therefore, illustrates the main qualitative difference introduced by L3: correlated-link damage can change both the damage record and the terminal outcome structure, even though the board state and outcomes remain classical.