Compression Efficiency and Structural Learning as a Computational Model of DLN Cognitive Stages

Alia Wu
Risk Efficacy & Redline Rising
wut08@nyu.edu
ORCID: 0009-0005-4424-102X

Abstract

We propose a computational instantiation of three cognitive stages from the Dot–Linear–Network (DLN) framework, grounded in a compression-efficiency thesis. DLN stages are characterized as representational topologies at the level of an agent's belief-dependency structure: Dot as no persistent belief graph (reactive policies with negligible state), Linear as a null graph over option beliefs (K independent option estimates with no information sharing), and Network as shared latent structure (a bipartite factor graph in which F latent factors connect to K options), augmented by a temporal exposure state and an explicit structural learning cycle (hypothesis $\rightarrow$ test $\rightarrow$ update/expand). In this mapping, Linear cognition learns and evaluates K options independently ($O(K)$ cognitive cost), while Network cognition represents shared structure once ($O(F)$ cognitive cost for F shared factors) and incorporates an explicit structural learning cycle: structural hypothesis $\rightarrow$ predictive test $\rightarrow$ update/expansion when the hypothesis fails. We distinguish two compression targets: *Option-factor structure* (shared structure in expected outcomes) and *Stakes-factor structure* (shared structure in consequence-bearing exposures). Their intersection yields *jointly efficient actions* that improve expected outcomes while simultaneously improving marginal stakes impact. In a bandit-like simulation with factor-structured options and factor-structured stakes (100 seeds, $K \in \{20, 50, 100, 200\}$, F=5, T=80), Network policies dominate Linear policies in cost-adjusted utility once the option set is sufficiently large (for example, at K=200 with stakes off, Network-Full achieves cost-adjusted utility 80.53 vs. Linear 64.46). With stakes active, Linear policies incur large exposure penalties because they ignore covariance-induced cross-terms from cumulative exposure, while Network policies learn factor-level exposures and track cumulative exposure, maintaining positive utility (for example, at K=200 with stakes on, mean utility 66.78 for Network vs. -929.11 for Linear). These results evaluate internal consistency of a DLN-to-computation mapping under explicit assumptions; they do not validate a developmental theory in humans.

1 Introduction

Individuals can possess comparable amounts of domain knowledge yet differ sharply in *how* that knowledge is organized and deployed. In practice, this manifests as differences in whether a person represents a situation as a set of disconnected facts, as a single privileged causal chain, or as an interacting system with feedback and cross-constraints. These differences matter for prediction, explanation, and decision support: two individuals can possess equivalent factual knowledge yet make systematically different choices because their internal representations differ in what they treat as shared, what they treat as independent, and how constraints propagate across elements. Making such structural differences explicit is relevant to human–AI interaction and cognitive modeling [21].

In related traditions, this emphasis on feedback and cross-constraints is central to systems thinking [27, 23].

We propose the Dot–Linear–Network (DLN) framework, which treats such differences as differences in *representational topology*: Dot as reactive processing without persistent belief structure, Linear as independent branch-by-branch estimation, and Network as shared latent structure with cross-branch information propagation. We employ graph-theoretic language: a cognitive state is represented as a graph $G = (V, E)$, where nodes are representational elements (concepts, features, action templates) and edges are statistical, causal, or constraint relations that allow information to propagate. Because "topology" can refer to multiple mathematical objects, we specify the relevant object precisely: our topological claims concern the agent's *belief-dependency structure* used to evaluate options. In this framework, Dot corresponds to no persistent belief structure (reactive baselines), Linear corresponds to a null graph over K option beliefs (independent option-by-option estimates with no edges), and Network corresponds to shared latent structure (options are d-connected through a smaller set of F latent factors). Temporal coupling through cumulative exposure and a meta-level revision loop create feedback at the level of *dynamics and learning*, even when the within-timestep belief graph is acyclic. This perspective complements existing work on knowledge graphs [13], classic semantic-network and spreading-activation models [4, 1], and cognitive architectures [2, 20] by emphasizing that agents can differ not only in what they know, but in the qualitative dependency structure of what they know.

This paper focuses on a narrow computational thesis: **Network representations are structurally more efficient in stable environments because they represent shared components once rather than redundantly across each branch.** The central claim does not concern exploration-driven adaptation. Exploration strategies are well-studied in bandits and reinforcement learning [9, 26, 29]; our focus is the representational *reuse* mechanism that can render Network efficient even in stable environments. Robustness and adaptation can follow from the same mechanism, but the core test here is an efficiency scaling claim: do policies that represent shared structure once avoid $O(K)$ growth in representational and computational requirements as the number of options K grows?

We decompose the thesis into two aligned compression targets. First, *option-factor structure*: many options share components in their expected outcomes, so learning can be amortized across options if shared structure is represented (hierarchical shrinkage provides a classical example [14, 6]). Second, *stakes-factor structure*: the consequences of actions often share common drivers (exposure factors) in addition to idiosyncratic effects. When consequences are coupled through shared drivers, marginal impact depends on cumulative exposure, so a decision policy must represent cross-terms (covariances) rather than treating each action's stakes in isolation. The intersection of these structures makes *jointly efficient actions* possible: a single action can be simultaneously beneficial for expected outcomes and beneficial for marginal stakes impact.

Mechanistically, we operationalize Network as a factorized model class coupled to an explicit structural learning cycle. The agent (i) begins with a compressed structural hypothesis, (ii) tests predictive adequacy against data, and (iii) expands or updates its hypothesis when it proves empirically inadequate. When the hypothesis is correct, the agent exploits compression and achieves $O(F)$-like scaling in the number of shared factors F; when the hypothesis fails, the agent pays additional cost to recover by expanding toward $O(K)$ state. This cycle constitutes the computational analogue of a structural prior $\rightarrow$ empirical test $\rightarrow$ revised prior mechanism.

Contributions. This paper contributes: (1) an explicit DLN-to-computation mapping centered on factorized compression rather than entropy-based exploration; (2) a stylized environment with

option-factor structure, stakes-factor structure, and an explicit cross-term that makes cumulative exposure matter; (3) scaling results with paired-seed statistical summaries (bootstrap confidence intervals and paired effect sizes) identifying where compression helps and where it does not; and (4) an explicit assumption audit and reproducible code.

We emphasize scope: the simulation evaluates internal consistency of a proposed computational interpretation; it does not establish that human cognitive development follows DLN.

2 DLN as a topological framework

We model a cognitive state as a graph $G = (V, E)$ where nodes are representational elements (features, concepts, action templates) and edges are relations (associations, causal links, constraints). This section formalizes DLN stage distinctions as topological regimes.

2.1 Topological regimes

DLN is often summarized as Dot $\rightarrow$ Linear $\rightarrow$ Network, with increasingly connected representational graphs. Topology claims require specification of the object whose topology is being characterized. In this paper, the object is the agent's *belief-dependency structure* used to evaluate options in a bandit-like setting (no explicit state transitions). The Linear stage thus corresponds to the simplest non-compressing representation: independent option-by-option estimation.

Dot stage (no persistent belief graph). Dot corresponds to policies with no persistent belief state for option evaluation. In our simulation, Dot selects uniformly at random and does not maintain outcome or exposure estimates. Topologically, this corresponds to an empty (or ephemeral) belief graph: no stored nodes or edges allow information to propagate across options.

Linear stage (independent option beliefs; null belief graph). Linear corresponds to maintaining K independent option beliefs (per-option means) with no edges between them. Topologically, this is a null graph on K belief nodes: no option shares an ancestor with any other option, so learning about option i does not update beliefs about option j. This captures the DLN property of branch-by-branch processing without reuse.

Network stage (shared latent structure; bipartite factor DAG). Network corresponds to representing options as arising from a smaller set of shared latent factors. In our instantiation, the within-timestep belief graph is a bipartite directed acyclic graph (DAG): F latent factor nodes connect to K option nodes. Options that share a factor share an ancestor and are therefore d-connected through the factor; evidence from one option updates the factor belief, which updates predicted values for all factor-mates. This shared-parent structure is the graph-theoretic source of integration in our implementation: shared structure represented once and reused across many options.

Where cycles enter: dynamics and meta-learning. Although the factor belief graph is a DAG, Network-stage cognition in DLN is also associated with feedback and revisitation. In this paper, those notions correspond to (i) a closed-loop dependency in the *system dynamics* when stakes are active (actions update exposure E_t, and E_t feeds back into the marginal penalty term used to evaluate future actions), and (ii) a *process-level* structural learning cycle (hypothesis $\rightarrow$ test $\rightarrow$ update/expand) that revises which belief graph is being used. This distinction is necessary to avoid

conflating graph topology with inference complexity: the present claim is not that cyclic belief graphs enable cheaper inference than trees. Rather, investing in shared latent structure can reduce the number of distinct quantities that must be learned from K to F when the structure hypothesis is correct.

2.2 Cost-scaling interpretation

The computational claim tested here is a scaling distinction:

- **Dot:** no accumulation of learning across encounters (reactive; baseline).
- **Linear:** independent estimation per option, yielding complexity that scales with the number of options K.
- **Network:** factorized representation that learns a smaller number of shared factors F and propagates factor updates to many options, yielding complexity that scales primarily with F rather than K.

Formal complexity statement. Let Linear maintain per-option sufficient statistics $(\widehat{\mu}_i, n_i)$ for $i \in \{1, \ldots, K\}$. Then its representational state requires

$$M_{\text{Lin}}(K) = \Theta(K).$$

In our implementation, Linear also scores all options each step, yielding per-decision computation $G_{\text{Lin}}(K) = \Theta(K)$. Let Network maintain per-factor statistics $(\widehat{m}_f, \widehat{b}_f, n_f)$ for $f \in \{1, \ldots, F\}$ plus the exposure state E_t, and use a bounded within-factor local search of size L (treated as a constant). Then

$$M_{\text{Net}}(F) = \Theta(F) + \Theta(1), \qquad G_{\text{Net}}(F) = \Theta(F + L) = \Theta(F) \text{ for fixed } L.$$

When the factor hypothesis fails and the model expands to a tabular representation, Network reverts to $\Theta(K)$ state and computation. Thus the compression thesis predicts an advantage only when $F \ll K$ *and* the factor hypothesis remains empirically adequate; when $K = F$ (no shared components), the advantage should vanish, constituting a falsification criterion for the compression thesis.

This relates to classical results on shrinkage and hierarchical estimation: when structure exists, sharing statistical strength across related items improves efficiency [14, 6]. In our mapping, Network operationalizes this by tracking factor-level beliefs and using them to score options, while retaining the capacity to expand to option-specific state when structure is violated.

2.3 Factor structure as a computational analogue of network integration

The computational instantiation used here is intentionally simple: Network is represented as a shared factor space. This is not meant to claim that human cognition is a literal factor model, but it renders the topology-to-computation mapping concrete. If we represent each option i as connected to a factor node c_i, then the representational graph becomes a bipartite structure in which many options share the same factor node. Updating the factor belief propagates immediately to all connected options. This is precisely the "represent shared components once" property that DLN attributes to integrated (network-stage) organization.

Stakes-factor structure adds an additional coupling: the current exposure state E_t links sequential decisions, and the marginal stakes impact contains a cross-term $2E_t b_i$. This introduces a feedback-like dependency across time: the consequences of an action depend on what has been selected

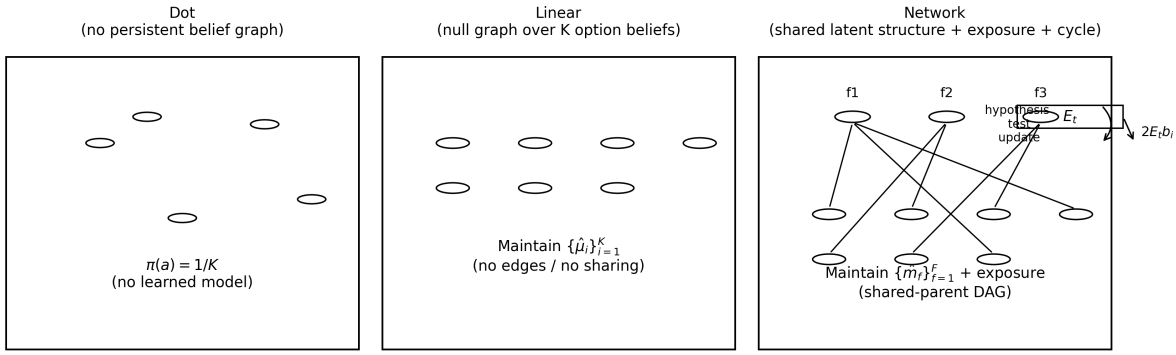

Figure 1: Belief-dependency structures in the present DLN instantiation. Dot maintains no persistent belief graph for option evaluation. Linear maintains K independent option beliefs (null graph; no edges). Network represents shared latent structure with F factor nodes connecting to K option nodes (bipartite DAG), tracks exposure E_t over time, and updates its structure via a hypothesis–test–update cycle.

before. A Linear representation that evaluates stakes per option without tracking E_t treats this dependency as if it were absent; a Network representation explicitly represents the coupling and therefore computes marginal impact. In this minimal setting, the coupling induces closed-loop decision dynamics: the state induced by prior actions feeds back into the evaluation of future actions.

This mapping implies a design requirement: to test DLN Network as integrated structure, the environment must contain shared components and cross-constraints that make reuse and coupling beneficial. If structure is absent or if $K = F$ (no shared components), the representation should not help. Those boundary tests are part of the falsifiable surface of the thesis.

3 Claims and predictions

3.1 Problem framing

We separate three claims.

(C1) Option-factor structure compression. Options share components. A Network-stage policy learns shared structure once and reuses it across many options (effective cost scaling with F), while a Linear-stage policy learns each option independently (effective cost scaling with K).

(C2) Stakes-factor structure compression. Consequence-bearing exposures share components. A Network-stage policy learns common exposure structure and tracks cumulative exposure so that a single hedge can reduce a shared driver once, rather than repeatedly per option. A Linear-stage policy treats exposures as independent and therefore misses covariance-driven marginal effects.

(C3) Intersection: jointly efficient actions. Actions can be simultaneously beneficial on expected outcomes (Option-factor structure) and beneficial on marginal exposure impact (Stakes-factor structure). A Network-stage policy can identify these overlaps because both decompositions are represented in a shared factor space.

3.2 Diagnostic signatures

The computational mapping yields diagnostic signatures that are testable in simulations (as demonstrated here) and, in principle, in behavioral experiments:

- **Scaling signature (reuse vs. redundancy).** In tasks with option-factor structure, regret or cost-adjusted utility should scale primarily with the number of factors F under Network representations but with the number of options K under Linear representations.
- **Marginal-stakes signature.** When consequences depend on cumulative exposure (covariance cross-terms), Network policies should behave as if evaluating *marginal* impact given current exposure state, while Linear policies behave as if evaluating each option's consequences independently.
- **Intersection signature (jointly efficient actions).** When option-factor and stakes-factor structure overlap, Network policies should identify actions that serve both objectives simultaneously more reliably than Linear policies, especially when the overlap is sparse.
- **Prior failure signature (recovery vs. silent degradation).** Under prior-wrong conditions, Network policies should exhibit measurable model updates (switches/expansions) and improved fit relative to no-test or no-update ablations, at the cost of increased cognitive cost.

These signatures separate the compression thesis from generic claims of adaptation or exploration. In particular, the scaling signature should obtain in stable environments without distribution shift.

3.3 Explicit assumptions

We state assumptions explicitly, as they determine the scope of claims the simulation can support.

- **A1 (observed option-factor labels).** Each option i has an observed option-factor label $c_i \in \{0, \ldots, F-1\}$. This stands in for a shared-component representation. (Linear agents choose to ignore it.)
- **A2 (observed stakes loading).** Each option exposes an observed stakes loading $b_i \in \mathbb{R}$.
- **A3 (structured outcomes; prior correct).** In the structured condition, mean outcomes follow a factor model: $\mu_i = m_{c_i} + \epsilon_i$.
- **A4 (unstructured outcomes; prior wrong).** In the unstructured condition, μ_i are independent and the label c_i is irrelevant for predicting outcome.
- **A5 (stakes objective and covariance cross-term).** Stakes are represented as a quadratic exposure objective in a 1D exposure state E_t. Selecting option i updates exposure $E_{t+1} = E_t + b_i$ and incurs incremental penalty

  $$\Delta P_t = \lambda \left((E_t + b_i)^2 - E_t^2 \right) = \lambda (2E_t b_i + b_i^2).$$

  The cross-term $2E_t b_i$ is the minimal mechanism by which cumulative exposure creates marginal effects that cannot be captured by option-by-option penalty terms alone.
- **A6 (cognitive cost proxy).** We use an algorithmic proxy cost consisting of memory units and compute units with weights (reported in the run manifest). This is a complexity surrogate, not a claim about biological energy.

Table 1: Assumption audit: what is built in, and relaxation paths.

ID	Assumption (informal)	Built-in?	Relaxation path
A1	Factor label c_i is observed (shared-component index).	Yes	Learn c_i / latent factors via clustering or representation learning.
A2	Stakes loading b_i is observed.	Yes	Learn b_i (or exposures) from observed consequences.
A3	Structured outcomes follow $\mu_i = m_{c_i} + \epsilon_i$.	Swept	Sweep structure strength (within-factor variance).
A4	Unstructured outcomes violate the factor prior (prior wrong).	Swept	Add partial structure and drifting structure regimes.
A5	Stakes penalty contains cross-term via exposure state E_t.	Yes	Move to multi-dimensional E_t and general quadratic forms.
A6	Cognitive cost is a proxy based on memory/compute scaling.	Yes	Sensitivity analysis; report Pareto frontiers instead of a single α.

4 Relationship to existing work

4.1 Cognitive architectures and bounded rationality

DLN is not intended to replace established cognitive theories or architectures, but to provide a structural lens that can complement them. Many cognitive architectures specify symbolic or production-system mechanisms [2, 20], and many modern AI systems operationalize relational knowledge as graphs [13]. The present contribution emphasizes that belief-dependency structure imposes computational constraints. When option beliefs are independent, evidence updates a single option; when options share latent structure, evidence propagates through shared parents, allowing one observation to update many options. This constitutes a representational and statistical-generalization claim rather than a claim about the computational cost of cyclic belief graphs.

DLN also connects to bounded rationality traditions in which limited computational resources necessitate simplified representations [28, 32, 11]. In this paper, the resource constraint is made explicit as a cognitive cost proxy and a scaling claim: Linear representations are expensive because they duplicate estimation across options, while Network representations can amortize learning across options when shared structure exists.

From a developmental perspective, stage theories often describe qualitative shifts in what operations become possible [25, 15]. The distinctive DLN claim is not about specific content operations, but about a shift in representational topology that makes certain computations cheaper (reuse) and makes others less likely (redundant branch-by-branch processing). Dynamic-systems

Table 2: Claim-to-test matrix: what each experiment condition identifies.

Claim	Condition	Observable signature
C1 (Option-factor compression)	Structured, stakes= 0, sweep K	Network surpasses Linear at large K in cost-adjusted utility; Linear cost grows with K while Network cost is $\approx O(F)$.
C2 (Stakes-factor compression)	Structured, stakes> 0, sweep K	Linear utility degrades substantially due to unmodeled exposure cross-terms; Network remains stable by tracking cumulative exposure and using marginal impact.
C3 (Intersection)	Structured, stakes> 0	Network selects jointly efficient actions (high outcome, favorable marginal stakes) at a higher rate than Linear, especially when overlap is sparse.
Recovery (prior wrong)	Unstructured, stakes= 0 (and > 0)	Network-Full shows switch/expansion events and outperforms NoTest/NoUpdate ablations, paying increased cognitive cost.
Boundary (no compression)	$K = F$	Network advantage disappears (falsifier for compression thesis).

accounts emphasize that qualitative shifts can emerge from continuous changes in coupling strength and feedback [30]; the structural learning cycle instantiated here is compatible with that view because it creates a mechanism by which representational coupling is strengthened when it explains data and relaxed when it fails.

4.2 Computational neuroscience: structure learning as a distinct operation

Computational neuroscience and cognitive science have repeatedly distinguished between (i) *structure learning* (selecting, constructing, or revising a latent representation such as latent causes, task sets, or structural forms) and (ii) *parameter learning* (updating values within a chosen representation). Gershman and Niv [10] review how inferring latent structure can support rapid generalization beyond what tabular reinforcement learning can provide, and Collins and Frank [5] model when learners create and reuse task-set structure rather than only fitting parameters within a fixed task representation. From a broader cognitive-science perspective, Kemp and Tenenbaum [16] demonstrate that humans can infer different *structural forms* from data, and hierarchical Bayesian accounts formalize when abstraction accelerates learning by sharing statistical strength across related instances [8]. Work on cognitive control and arbitration similarly emphasizes cost–benefit tradeoffs in deploying more computationally demanding strategies, including when model-based control becomes worthwhile [18].

Neural evidence is consistent with this two-level distinction. Causal and structure learning signals have been reported in frontoparietal networks that are at least partially dissociable from striatal value-learning signals [31]. Related work suggests that frontopolar and rostrolateral prefrontal cortex can compute reliability over alternative strategies or learning systems [22] and can support reasoning over counterfactual strategies not currently deployed [17]. We do not claim a neural implementation in this paper; we cite these results only to motivate treating structure learning as a distinct computational operation rather than folding it into parameter learning.

Building on this foundation, our contribution is more circumscribed: we instantiate DLN stage differences as differences in *belief-dependency structure* and derive a concrete efficiency prediction for

a particular representational family. First, we make the scaling implication of shared latent structure explicit as an $O(F)$ versus $O(K)$ crossover (Section 2.2), where F is the number of shared factors and K is the number of options. Second, we extend the same compression logic to *stakes-factor structure* by introducing an exposure objective with an explicit covariance cross-term, so that the marginal consequence of an action depends on cumulative exposure state. Third, we connect these representational regimes to a three-stage DLN mapping (Dot/Linear/Network) and test falsifiers and ablations (e.g., $K{=}F$ and prior-wrong conditions) that isolate when the structural learning cycle matters.

This positioning also clarifies how our claims relate to graph-theoretic complexity results. Classical work on probabilistic graphical models [24] and the treewidth / tractable-cognition literature [33, 19] analyze the complexity of inference *within a fixed model structure* and establish that low-treewidth graphs enable efficient inference. The within-timestep belief graph for the Network agent is itself a low-treewidth bipartite DAG; we do not claim that cycles in belief graphs make inference cheaper than trees. Instead, we focus on a representation-construction question: when the world admits shared structure with $F \ll K$ and the factor hypothesis is empirically adequate, representing shared structure reduces the number of distinct quantities that must be learned from K to F; when structure is absent or incorrect, expansion toward a tabular representation removes the advantage.

5 Simulation design

5.1 Structural learning cycle

The Network stage is not defined merely by the presence or absence of structure. It is defined as an explicit learning cycle:

1. **Structural hypothesis (Bayesian prior over structure).** Start with a compressed factor hypothesis for outcome prediction.
2. **Predictive test (frequentist check).** Compare rolling predictive error of the factor predictor against an expanded tabular predictor.
3. **Branch.** If the factor model predicts well, exploit compression ($O(F)$ state). If it predicts poorly, expand/update to a tabular model ($O(K)$ state).

This constitutes a concrete implementation of a structural prior $\rightarrow$ test $\rightarrow$ revised prior mechanism: the hypothesis is the factor predictor, the test is a rolling-window predictive check, and the revised hypothesis is represented by switching or expanding model class when the hypothesis fails.

5.2 Environment

We use a bandit-like environment with K options and F factors. Each option i carries a factor label c_i. Outcomes are Gaussian:

$$r_t \sim \mathcal{N}(\mu_{a_t}, \sigma_r^2).$$

In the structured (prior-correct) condition, $\mu_i = m_{c_i} + \epsilon_i$ with small idiosyncratic noise; in the unstructured (prior-wrong) condition, the μ_i are independent. Stakes loadings are factor-structured:

$$b_i = g_{c_i} + \delta_i,$$

with small idiosyncratic noise. Utility is the sum of outcomes minus incremental stakes penalties.

5.3 Agents

Dot. The Dot agent selects uniformly at random and performs no learning.

Linear. The Linear agent maintains tabular frequentist mean estimates per option; action selection scores each option using a local stakes heuristic that ignores the cross-term $2E_t b_i$:

$$\text{score}(i) = \widehat{\mu}_i - \lambda b_i^2.$$

This scoring function treats hedging as separable across options and does not track cumulative exposure.[1]

Network-Full. The Network-Full agent maintains factor-level outcome estimates $\widehat{m}_f$ and factor-level exposure estimates $\widehat{b}_f$, tracks E_t, and uses the true marginal penalty form. Within a chosen factor, Network searches over at most L=5 options to avoid hiding $O(K)$ scans in factor selection. It also runs the predictive check and can expand to tabular outcome estimates when the factor hypothesis fails.

Ablations. Network-NoTest disables the predictive check; Network-NoUpdate runs the check but disallows switching.

5.4 Metrics and statistics

Per episode we report: (i) total outcome, (ii) total stakes penalty, (iii) utility $U = \sum_t (r_t - \Delta P_t)$, (iv) cognitive cost proxy C, (v) cost-adjusted utility $U - \alpha C$, (vi) jointly efficient action pick count, and (vii) number of hypothesis expansions (switches).

Results are averaged over 100 random seeds with paired comparisons across seeds (same environment seed per agent). For key paired differences we report bootstrap 95% confidence intervals (percentile bootstrap over seeds) and paired Cohen's d on the seed-wise difference.

6 Results

6.1 Option-factor structure compression (Claim C1)

Table 3 reports cost-adjusted utility in the stable structured environment (stakes off). As K grows, Linear cognitive cost scales with K, while Network cost is approximately constant in F (Table 4). Network overtakes Linear for larger option sets: at $K = 200$, Network-Full achieves cost-adjusted utility 80.53 vs. Linear 64.46 (paired difference mean +16.08).

6.2 Stakes-factor structure compression (Claim C2)

With stakes active, Linear agents incur large penalties because they do not represent the cross-term $2E_t b_i$ and cannot hedge cumulative exposure. Network agents track E_t and learn factor-level exposure structure, enabling hedging and keeping penalty bounded. At $K = 200$ in the structured setting, Network-Full maintains positive mean utility (66.78) while Linear incurs substantial negative utility (-929.11). The paired utility difference is +995.89.

[1] A hybrid "Linear+E_t" agent could track cumulative exposure while maintaining independent option beliefs. We exclude this design because tracking E_t requires representing how past choices affect future evaluations—a cross-option temporal dependency inconsistent with the branch-by-branch independence that characterizes Linear-stage cognition. Isolating the factor-compression and exposure-tracking contributions via such an ablation is a direction for future work.

6.3 Dot baseline

Dot serves as the null-graph baseline: it does not accumulate value estimates, does not represent option-factor or stakes-factor structure, and does not track cumulative exposure E_t. As a result, Dot does not benefit from increasing K even when option-factor structure is present. In the stable structured setting with stakes off (Table 3), Dot's cost-adjusted utility remains near zero across K (means between -0.51 and 0.88), reflecting near-chance outcomes combined with near-zero cognitive cost.

With stakes active (Table 5), Dot incurs negative utility (roughly -57 to -109) because cumulative exposure follows an uncontrolled random walk. Dot avoids Linear's extreme degradation only because it also avoids systematic exploitation; this reflects robustness through non-commitment rather than compression. Network differs by maintaining positive utility while simultaneously optimizing outcomes and controlling marginal exposure impact.

6.4 Jointly efficient actions (Claim C3)

In the structured stakes condition, the environment contains a sparse set of actions that are *jointly efficient* by construction: they are high on expected outcome *and* reduce marginal exposure impact under the stakes objective. Formally, the true incremental penalty for selecting option i at exposure state E_t is

$$\Delta P_t(i) = \lambda \left((E_t + b_i)^2 - E_t^2\right) = \lambda(2E_t b_i + b_i^2),$$

so the *marginal* stakes contribution depends on the sign interaction between the cumulative exposure E_t and the option's loading b_i. A jointly efficient action is one that simultaneously has high μ_i and (when E_t is positive) has $b_i < 0$ so that the cross-term $2E_t b_i$ is negative.

Evaluation definition. At environment initialization we flag a small subset of options as jointly efficient (the `dual_mask`): options in the top reward quantile that also have negative exposure loading. This flag is computed from the environment's ground-truth (μ_i, b_i) and is fixed *before* any agent acts. We report the pick rate as the fraction of timesteps in which the chosen action lies in this flagged set (Figure 4).

Why intersection requires representing both decompositions. To identify a jointly efficient action, a policy must (i) know which actions are promising on expected outcomes (Option-factor structure) and (ii) evaluate *marginal* stakes impact given current exposure E_t (Stakes-factor structure). Network-Full represents both: it learns factor-level outcome estimates, learns factor-level exposure estimates, tracks E_t, and scores factors/actions using the true marginal form. Linear fails on (ii): it uses a local stakes heuristic $\widehat{\mu}_i - \lambda b_i^2$ that ignores the cross-term and the sign interaction with E_t, so it systematically undervalues actions that are beneficial precisely because their exposure loading offsets existing exposure. Dot fails on both (i) and (ii) because it is non-learning and exposure-agnostic.

6.5 Prior-wrong condition: the learning cycle expands the model

In the unstructured condition, the factor hypothesis for outcome prediction is incorrect. Network-Full allocates tabular state (expansion) and improves raw utility relative to NoTest/NoUpdate ablations, while paying an explicit cognitive cost for doing so. For example at $K = 200$ (unstructured, stakes off), Network-Full exceeds Network-NoTest by $+7.31$ utility (bootstrap 95% CI $[2.23, 12.66]$)

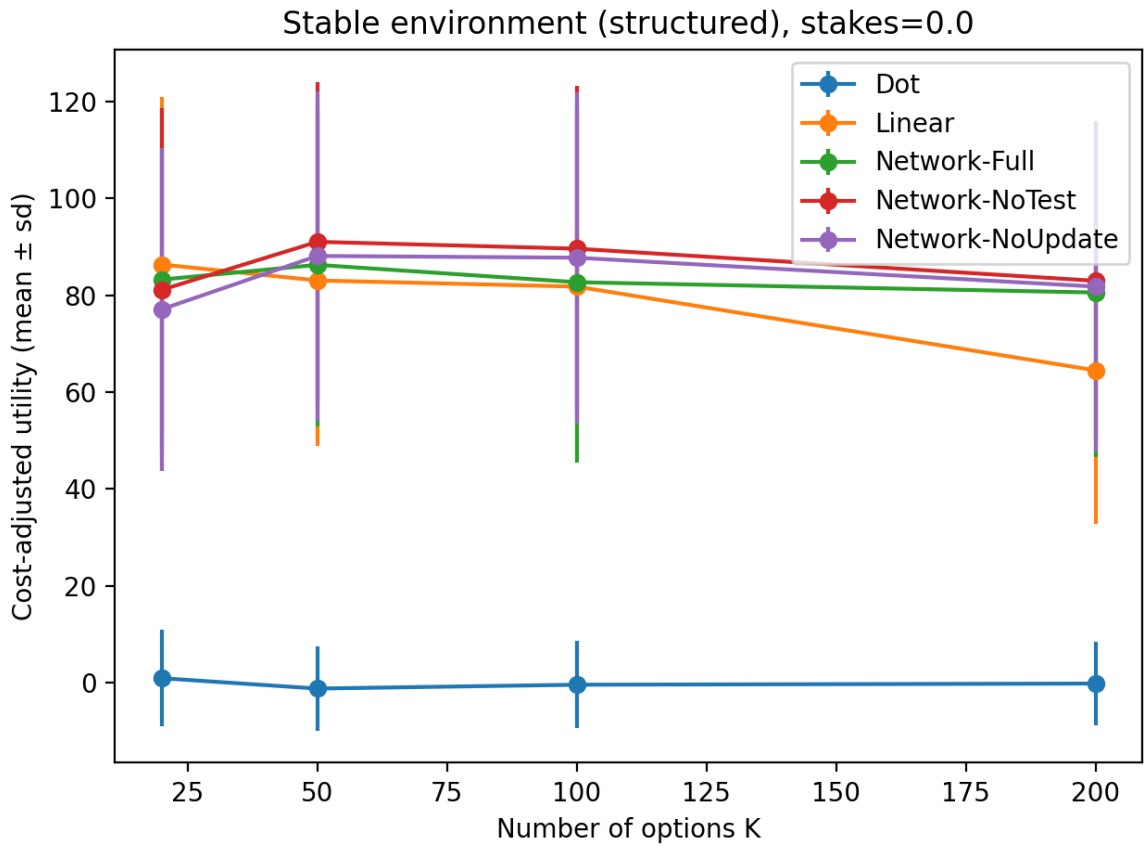

Figure 2: Stable structured environment (stakes=0): cost-adjusted utility vs. number of options K for Dot, Linear, and Network-Full.

and exceeds Network-NoUpdate by +8.50 (bootstrap 95% CI $[3.65, 13.54]$), while exhibiting more hypothesis expansions (Figure 5).

Interpreting the ablations. Network-NoTest corresponds to compression without validation: it trusts the factor hypothesis unconditionally and therefore continues pooling across irrelevant labels when the prior is incorrect, accumulating systematic bias. Network-NoUpdate corresponds to validation without adaptation: it allocates computation (and may even allocate tabular state for comparison) to detect mismatch, but is not permitted to switch its action model, so it pays overhead without recovering performance. Network-Full requires both the predictive test and the update/expansion branch to recover under prior failure; the performance gaps in the unstructured condition demonstrate that the *learning cycle* (not just factorization) is performing work.

7 Discussion

7.1 Interpretation

Under Assumptions A1–A6, the simulation isolates the efficiency consequences of representing shared structure. Dot is included as the null-graph baseline (no accumulation, no compression) to anchor the three-stage ordering; it is expected to ignore both structure and stakes. The simulation does

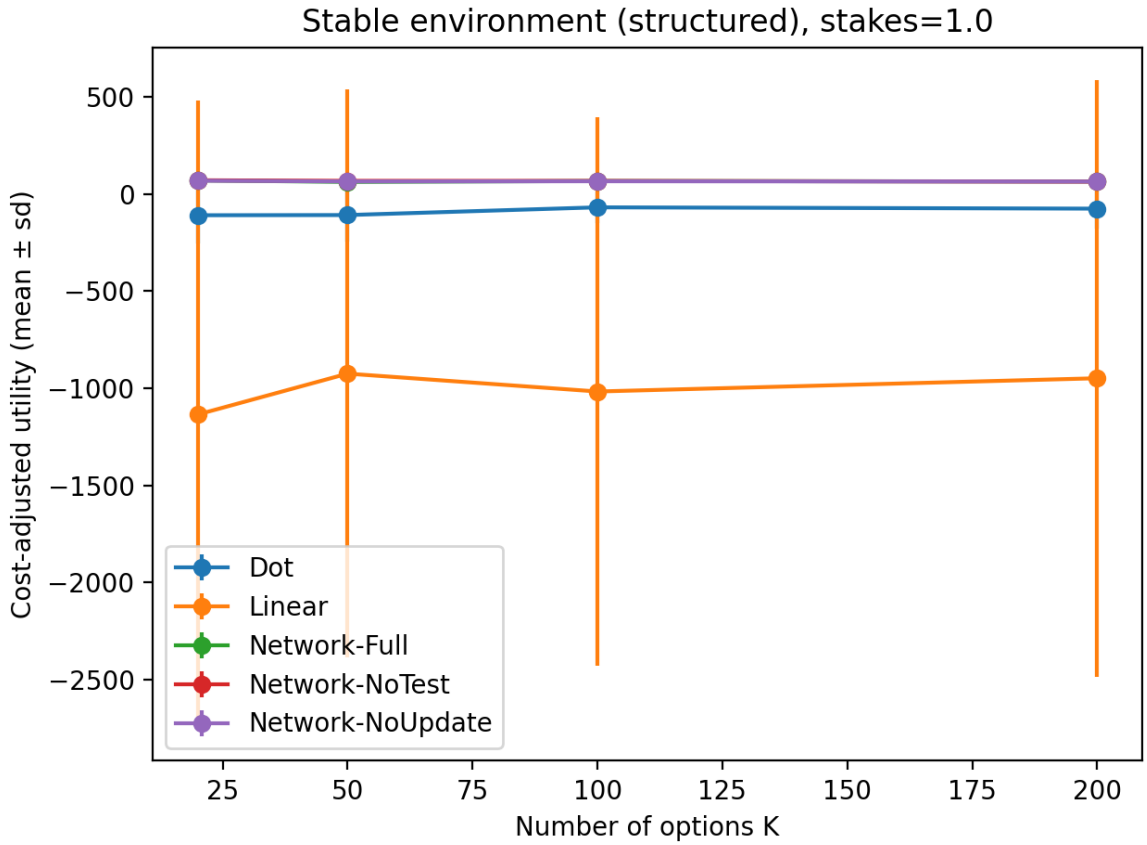

Figure 3: Structured environment with stakes (stakes=1): cost-adjusted utility vs. K. Linear incurs substantial negative utility due to unmodeled cumulative exposure; Network maintains utility by tracking E_t and using marginal impact.

not establish whether humans actually represent options and exposures in this way. Developmental interpretations are also potentially culturally contingent; see WEIRD-sampling cautions [12]. In humans, metacognitive monitoring and perceived agency may modulate when higher-cost integrated reasoning is deployed [7, 3], but we do not model that deployment layer here. The mapping should be interpreted as follows: *if* DLN Network is interpreted as a topology that supports factor reuse and cross-branch interaction, *then* it should produce $O(F)$-like scaling advantages and the ability to identify jointly efficient actions.

The compression-efficiency thesis tested here is distinct from exploration-driven adaptation; the Network advantage obtains in stable environments without requiring distribution shift or novelty-seeking (Section 1). Exploration mechanisms are well-studied elsewhere [9, 26]; our focus is the representational reuse that makes compression beneficial even when the environment is stationary.

7.2 Boundary conditions

The results depend on explicit structural gaps between K and F and on the presence of cross-terms in the stakes objective.

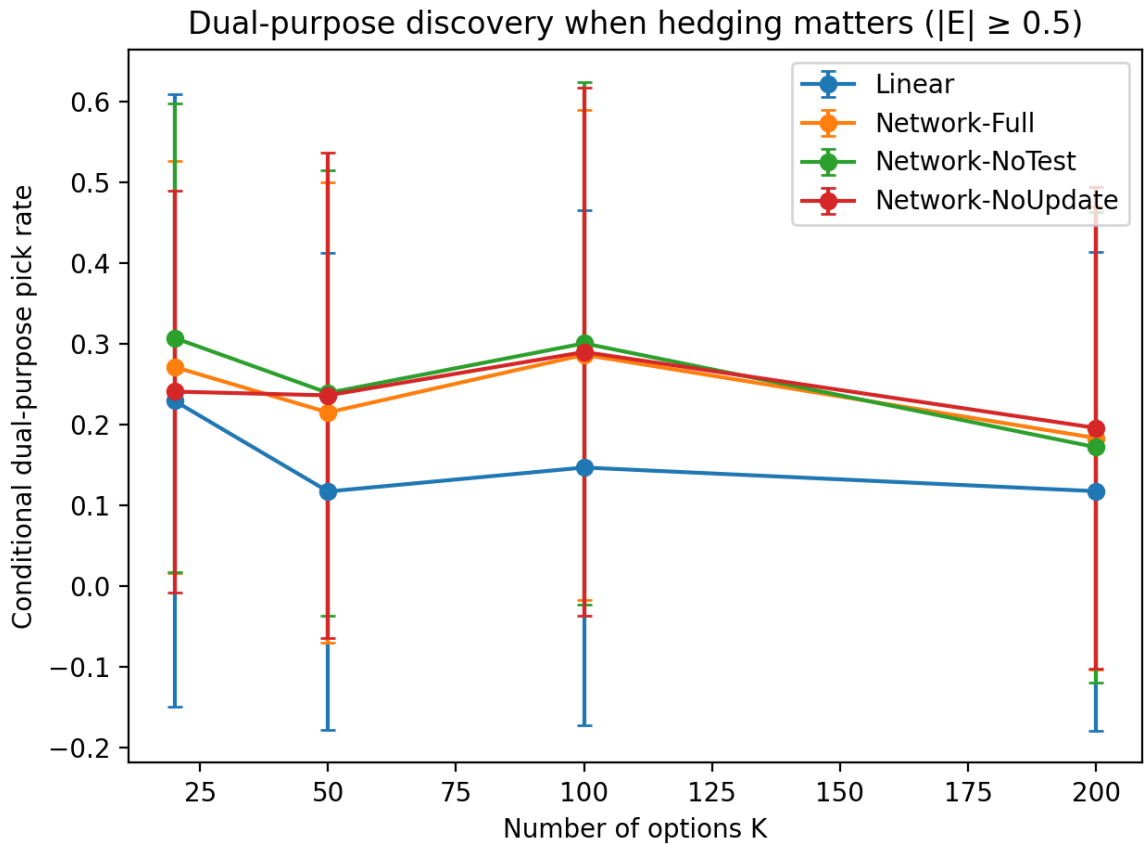

Figure 4: Jointly efficient action pick rate (structured, stakes=1).

When $K = F$ (no compression available). If every option is its own factor, then there is no shared component to compress. In that case, a Network factor representation should offer no advantage over Linear, and may perform worse due to representational overhead. This constitutes a key falsifier for the compression thesis.

Partial structure. If option-factor structure is weak (large within-factor deviations), the benefit of shrinkage and factor scoring should degrade smoothly. Likewise for stakes-factor structure: if exposures are not aligned with factors, the hedging benefit should degrade. A structure-strength sweep makes these boundary conditions visible.

Stakes scaling versus stakes structure. This paper models stakes through *structured exposures* with a covariance-inducing cross-term. Scaling stakes magnitude alone is a different question; both can be studied, but they test different mechanisms.

7.3 Limitations and future work

Observed factor labels (A1). We assume the Network agent has access to a shared-component representation c_i. This provides a representational advantage rather than requiring the agent to learn it. *Future direction:* replace c_i with learned clustering or learned latent factors (Bayesian

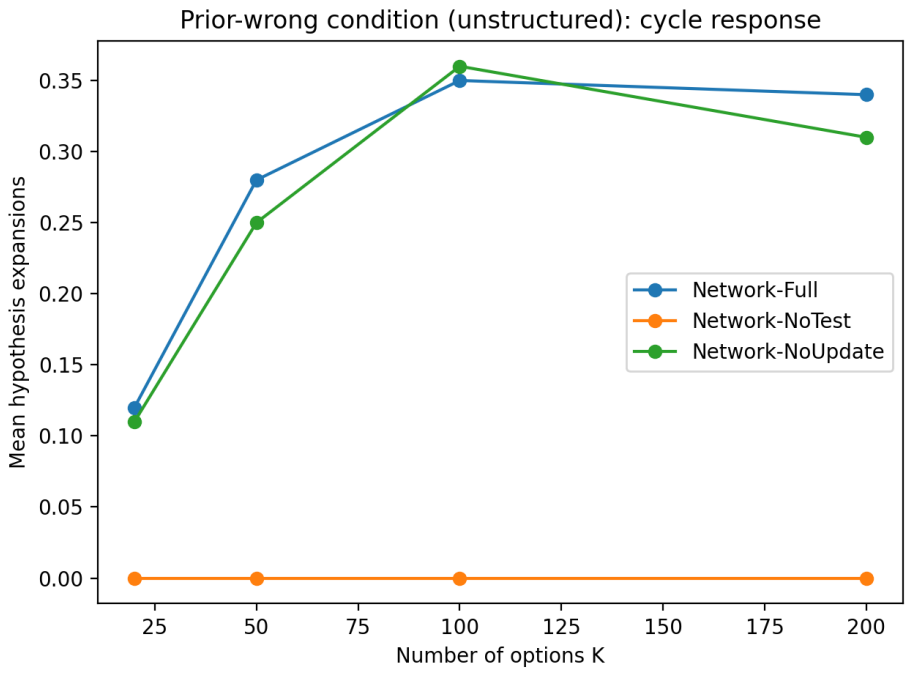

Figure 5: Prior-wrong condition (unstructured, stakes=0): mean number of hypothesis expansions (switches) for Network variants.

Table 3: Cost-adjusted utility (mean ± sd) in the stable structured environment (stakes=0).

K	Dot	Linear	Network-Full
20	0.88 ± 9.99	86.31 ± 34.59	83.21 ± 35.27
50	-1.25 ± 8.64	83.03 ± 34.07	86.25 ± 33.41
100	-0.46 ± 9.03	81.76 ± 35.60	82.67 ± 37.32
200	-0.22 ± 8.69	64.46 ± 31.73	80.53 ± 33.90

nonparametrics or representation learning) and report the additional sample and cognitive costs required to learn the representation.

1D exposure state (A5). We use a single exposure dimension to make the cross-term explicit and interpretable. *Future direction:* extend to d-dimensional exposures with $E_t \in \mathbb{R}^d$ and penalty $\|E_t\|^2$ (or a quadratic form $E_t^\top \Sigma E_t$). Measure how Network cost scales with the number of exposure factors rather than with the number of options.

Cognitive cost proxy (A6). We use an algorithmic proxy (memory and compute scaling). *Future direction:* perform sensitivity analysis over cost weights and α, and report Pareto frontiers over (utility, cost) rather than a single calibrated point estimate.

Local search within factor. Network uses a bounded local search of size L within a chosen factor to avoid hiding $O(K)$ scans. *Future direction:* vary L (ablation) to quantify the tradeoff between factor-level compression and within-factor deviation exploitation.

Table 4: Cognitive cost proxy (mean ± sd) showing $O(K)$ scaling for Linear and $O(F)$ scaling for Network.

K	Linear	Network-Full
20	2.27 ± 0.04	1.68 ± 0.07
50	5.07 ± 0.13	1.75 ± 0.02
100	9.73 ± 0.25	1.75 ± 0.02
200	19.05 ± 0.50	1.75 ± 0.02

Table 5: Utility (mean ± sd) in the structured environment with stakes (stakes=1).

K	Dot	Linear	Network-Full
20	−108.65 ± 145.49	−1132.80 ± 1616.31	72.28 ± 35.44
50	−107.71 ± 138.95	−919.45 ± 1462.16	64.65 ± 34.10
100	−68.02 ± 83.13	−1006.74 ± 1412.20	69.18 ± 33.07
200	−74.77 ± 101.39	−929.11 ± 1537.36	66.78 ± 29.62

Decomposing compression and exposure tracking. The present design confounds two mechanisms: factor-level compression (representing shared structure once) and cumulative exposure tracking (using E_t to compute marginal impact). A Linear agent augmented with E_t-tracking but no factor structure would isolate these contributions. *Future direction:* implement a "Linear+E_t" baseline and measure how much of the stakes-condition advantage (Claim C2) arises from compression versus exposure tracking.

8 Reproducibility

All tables in this paper are derived from `outputs/paper/results/episode_metrics.csv`. The default run that reproduces the CSV and baseline figures is:

```
python src/dln_core_variable_cycle.py --preset paper --out outputs/paper
```

The run writes `outputs/paper/results/manifest.json` containing hyperparameters, seeds, and file hashes. Baseline figures are written under `outputs/paper/artifacts/figures/`. To regenerate figures with different styling, use the CSV as the single source of truth and rebuild plots externally.

Code availability. The complete source code, simulation framework, and reproducibility scripts are available at: `https://github.com/aliawu08/dln-compression-model`

9 Conclusion

We presented a computational mapping of DLN stages where the Network stage is defined by compression efficiency and an explicit structural learning cycle rather than by entropy-based adaptivity. Dot remains a baseline regime with no accumulation: it neither benefits from option-factor structure nor controls cumulative exposure under stakes, and therefore stays near baseline utility. Under explicit assumptions, Network policies outperform Linear policies in stable environments once

option sets are sufficiently large (Option-factor structure compression), avoid exposure degradation under stakes through factor-level exposure learning and cumulative exposure tracking (Stakes-factor structure compression), and expand/update when a structural hypothesis fails (learning cycle). The mapping is falsifiable at the level of assumptions and boundary tests: when shared structure vanishes ($K = F$) or when structure is weak, Network advantage should disappear. This provides a concrete bridge between qualitative DLN descriptions and measurable computational predictions.

References

[1] J. R. Anderson. *The Architecture of Cognition*. Harvard University Press, 1983.

[2] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin. An integrated theory of the mind. *Psychological Review*, 111(4):1036–1060, 2004.

[3] A. Bandura. *Self-Efficacy: The Exercise of Control*. W. H. Freeman, 1997.

[4] A. M. Collins and E. F. Loftus. A spreading-activation theory of semantic processing. *Psychological Review*, 82(6):407–428, 1975.

[5] A. G. E. Collins and M. J. Frank. Cognitive control over learning: creating, clustering, and generalizing task-set structure. *Psychological Review*, 120(1):190–229, 2013.

[6] B. Efron and C. Morris. Stein's paradox in statistics. *Scientific American*, 236(5):119–127, 1977.

[7] J. H. Flavell. Metacognition and cognitive monitoring: a new area of cognitive-developmental inquiry. *American Psychologist*, 34(10):906–911, 1979.

[8] S. J. Gershman. On the blessing of abstraction. *The Quarterly Journal of Experimental Psychology*, 70(3):361–365, 2017.

[9] S. J. Gershman. Deconstructing the human algorithms for exploration. *Cognition*, 173:34–42, 2018.

[10] S. J. Gershman and Y. Niv. Learning latent structure: carving nature at its joints. *Current Opinion in Neurobiology*, 20(2):251–256, 2010.

[11] G. Gigerenzer and D. G. Goldstein. Reasoning the fast and frugal way: models of bounded rationality. *Psychological Review*, 103(4):650–669, 1996.

[12] J. Henrich, S. J. Heine, and A. Norenzayan. The weirdest people in the world? *Behavioral and Brain Sciences*, 33(2–3):61–83, 2010.

[13] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, *et al.* Knowledge graphs. *ACM Computing Surveys*, 54(4):1–37, 2021.

[14] W. James and C. Stein. Estimation with quadratic loss. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, pages 361–379. University of California Press, 1961.

[15] R. Kegan. *In Over Our Heads: The Mental Demands of Modern Life*. Harvard University Press, 1994.

[16] C. Kemp and J. B. Tenenbaum. The discovery of structural form. *Proceedings of the National Academy of Sciences*, 105(31):10687–10692, 2008.

[17] E. Koechlin. An evolutionary computational theory of prefrontal executive function in decision-making. *Philosophical Transactions of the Royal Society B*, 369(1655):20130474, 2014.

[18] W. Kool, F. A. Cushman, and S. J. Gershman. When does model-based control pay off? *PLoS Computational Biology*, 12(8):e1005090, 2016.

[19] J. Kwisthout, T. Wareham, and I. van Rooij. The necessity of bounded treewidth for efficient inference in Bayesian networks. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI)*, pages 237–242, 2010.

[20] J. E. Laird. *The Soar Cognitive Architecture*. MIT Press, 2012.

[21] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017.

[22] S. W. Lee, S. Shimojo, and J. P. O'Doherty. Neural computations underlying arbitration between model-based and model-free learning. *Neuron*, 81(3):687–699, 2014.

[23] D. H. Meadows. *Thinking in Systems: A Primer*. Chelsea Green Publishing, 2008.

[24] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[25] J. Piaget. *The Origins of Intelligence in Children*. International Universities Press, 1952.

[26] D. Russo, B. Van Roy, A. Kazerouni, I. Osband, and Z. Wen. A tutorial on Thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96, 2018.

[27] P. M. Senge. *The Fifth Discipline: The Art and Practice of the Learning Organization*. Doubleday, 1990.

[28] H. A. Simon. A behavioral model of rational choice. *The Quarterly Journal of Economics*, 69(1):99–118, 1955.

[29] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

[30] E. Thelen and L. B. Smith. *A Dynamic Systems Approach to the Development of Cognition and Action*. MIT Press, 1994.

[31] M. S. Tomov, H. M. Dorfman, and S. J. Gershman. Neural computations underlying causal structure learning. *Journal of Neuroscience*, 38(32):7143–7157, 2018.

[32] A. Tversky and D. Kahneman. Judgment under uncertainty: heuristics and biases. *Science*, 185(4157):1124–1131, 1974.

[33] I. van Rooij. The tractable cognition thesis. *Cognitive Science*, 32(6):939–984, 2008.