Policies, Planning, and Feedback Control

Part V of *An Algebra of Agents*

Matthew Long
The YonedaAI Collaboration, YonedaAI Research Collective
Chicago, IL
matthew@yonedaai.com

September 2026

Abstract

Artificial intelligence and control present decision making in separate formalisms: Markov decision processes for reinforcement learning, AND/OR transition systems for planning under nondeterminism, belief states for partial observability, safety games for shields, and difference equations for feedback control. This Part reads each as a term of the algebra of agents fixed in earlier Parts of this series and proves properties of the resulting closed loops. No operator is added. A policy constrains the policy component of an agent's step function; an option is a guarded recursion; a shielded policy is one guarded choice inserted into an unshielded loop; a plan is a term over sequencing, guarded choice and recursion; a linear controller is a closed loop over the identity monad. The closed loop of the algebra carries a one-tick register, so a memoryless policy and a memoryless environment interact on a three-tick cycle, and every result here concerns that epoch process. Within it the discounted value of the closed loop is the value of the policy in the Markov decision process, and value composes along sequential composition at a stopping time. A finite partially observable process induces a belief agent whose closed-loop value is the value of the belief process. A shield built from a controllable invariant preserves the invariant for every policy, a human approval gate agrees with the shield whose fallback is idling up to stuttering, and the linear closed loop converges exactly when its spectral radius is below one.

1 Introduction

At least five vocabularies describe an agent choosing its next action, and they do not talk to each other. A reinforcement learner has a policy, a value function and a discount factor. A planner has a domain, a goal and a solution that is weak, strong or strong cyclic. A robot has a plant, a gain matrix and a spectral radius. A safety engineer has a specification, a winning region and a shield. A language model application has a prompt, a tool call and a loop that stops when the model says it is done. Each vocabulary carries its own notion of what the system is, so a statement proved in one of them does not transfer to another even when the underlying construction is the same.

This series fixes one object and one set of operators and asks which of these constructions are terms over them. Part I fixes the object: an agent is an effectful Mealy coalgebra with typed termination, running in discrete time over a commutative effect monad. Parts II, III and IV add operators with their laws: sequential composition, choice, synchronous product, recovery and guarded recursion; delayed feedback, channels, asynchronous interleaving and the human approval gate; supervision, checkpointing and replay. This Part adds nothing. Every construction below is a

term over those operators, and every result is a statement about the closed-loop semantics of such a term.

The central claim of the series is the following.

> Part I fixes the object: discrete time, an effect monad T drawn from the list Id, $\mathcal{P}$, $\mathcal{D}$, and agents as effectful Mealy coalgebras with typed termination and failure as a terminal outcome. Parts II to IV introduce the operators under additional assumptions stated in each Part and prove, for each operator, that it is well-defined on agents, which relation it respects, and which laws hold at that relation: Part II for ;, $\oplus$, $\otimes$, $\triangleright$ and μ; Part III for Tr, channels, $\|$ and Gate_H; Part IV for Sup, Persist and Replay; Part V introduces no operator and interprets policies, plans, options, shields and controllers as terms of Parts I to IV, proving properties of their closed loops. For each of sixteen fields the series gives an explicit translation of one named mechanism into a term of the algebra together with an adequacy result for that translation, which is a soundness theorem, a one-way simulation, or a proved limitation; no field is claimed to reduce to the algebra.

For six mechanisms this Part proves the closed-loop half of that sentence. It adds five assumptions: the interaction is epoch aligned in the sense of Section 4; the Markov and partially observable models are finite with rational parameters and a discount $\gamma \in [0, 1)$; option durations have finite expectation; the shielded predicate is controllable; and the plant is a discrete-time linear system with real matrices.

Part I's closed loop imposes a delay. It passes each side's output to the other through a one-tick register, so an agent never sees the consequence of its own action in the tick in which it acts. A memoryless policy paired with a memoryless environment then settles into a three-tick cycle: the environment applies the previous action, the agent reads the resulting announcement, and the agent acts. Section 4 identifies the induced epoch process with the one-step process the textbook models describe, and every result after that section is a statement about the epoch process. The delay is not an artifact to be quotiented away. Section 11 exhibits gains that stabilize the aligned loop and destabilize the loop with one extra epoch of delay.

A tool-calling loop over a language model with a bounded context and a bounded generation length is an agent over the distribution monad. The ReAct loop is the guarded recursion $\mu X.(((\mathsf{Think}; \mathsf{Act}) ; X) \oplus_{\mathrm{done}} \mathsf{skip})$, and it is bisimilar to the loop one would write by hand.

The discounted value of a closed loop against a finite Markov decision environment is the value of the policy in that decision process, so policy evaluation and policy improvement become operations on agents.

Value composes along sequential composition at the halting epoch. This is the semi-Markov equation of the options framework, and it holds whenever the halting epoch has finite expectation.

A finite partially observable process induces an agent whose state is a rational belief and whose closed-loop value is the value of the belief process.

A shield built from a controllable predicate is a single guarded choice inserted into the policy loop, and it preserves the predicate for every policy: surely against a nondeterministic environment, and on every positive-probability path against a stochastic one.

A human approval gate has the same marker-erased, stutter-collapsed traces as the shield whose fallback is idling. That is why it needs no controllability assumption and why it cannot deliver progress.

The weak, strong and strong cyclic solutions of nondeterministic planning are the existential-path, winning-strategy and fair-winning solutions of the closed-loop game, and the strong cyclic solutions are the guarded recursions that win under outcome fairness.

Table 1: Objects imported by this Part, with the Part and numbered statement in which they are defined or proved, and the short form of the label they carry there. A short form such as `interface` abbreviates the full label `def:agent-object:interface`, whose kind prefix and Part slug are determined by the other two columns. Nothing in this table is redefined here.

Object	Where	Label
Interface (I, Σ) with idle symbols ε and τ	I, Def. 2.1	`interface`
Effect monad T and the instances Id, $\mathcal{P}$, $\mathcal{D}$	I, Def. 2.2	`effect-monad`
Agent $(S, \mathrm{init}, \mathrm{step}, \mathrm{halt})$ of type $X \Rightarrow Y$	I, Def. 2.10	`agent`
Policy and update factorization (π, δ) of a step	I, Def. 2.13	`policy-update`
Trace semantics $[\![A]\!]$ and trace equivalence $\simeq_{tr}$	I, Defs. 3.2, 3.3	`trace`, `trace-equiv`
Bisimilarity $\sim$; bisimilarity implies trace equivalence	I, Def. 4.1, Thm. 4.4	`bisim`, `bisim-implies-trace`
Stuttering refinement $\sqsubseteq$	I, Def. 5.4	`refinement`
Reachable set; invariant; greatest invariant	I, Defs. 8.1, 8.2, Thm. 8.3	`reachable`, `invariant`, `greatest-invariant`
Environment co-agent and closed loop $\mathrm{Run}(A, E)$	I, Defs. 9.1, 9.2	`environment`, `closed-loop`
The closed loop is a Markov chain for finite $\mathcal{D}$	I, Prop. 9.3	`closed-loop-markov`
Identity agent skip_X; delay delay_d	II, Defs. 4.1, 4.4	`skip`, `delay`
Sequential composition ; and the category $\mathrm{Ag}(I, \Sigma)$	II, Def. 4.6, Thm. 6.1	`seq`, `sequential-category`
Tests and guarded choice $\oplus_b$	II, Defs. 4.9, 4.10	`test`, `guarded-choice`
Guardedness, guarded recursion and the derived loop	II, Defs. 4.19, 4.20, 4.22	`guarded`, `mu`, `while`
Bisimilarity is a congruence for the operators	II, Thm. 5.1	`congruence`
Unique solutions of guarded equations	II, Thm. 7.1	`unique-solutions`
Human co-agent and the approval gate Gate_H	III, Defs. 8.1, 8.2	`human-coagent`, `gate`
Gate refinement under an always-approving oracle	III, Thm. 8.3	`gate-refinement`
Replay determinism for a completely logged run	IV, Thm. 7.3	`replay-determinism`

The discrete linear closed loop converges from every initial state exactly when the spectral radius of $F - GK$ is below one.

The results are restricted to fixed policies, to discrete time, and to idealized inference. A change in the parameters of a policy produces a different agent, so learning is an operation on agents rather than a term of the algebra, and no theorem here concerns it. Part I's commitment to discrete time confines the linear results to sampled systems. The abstraction of a language model used in Section 5 governs generation from a bounded context and excludes the batch-dependent nondeterminism of a production serving stack.

2 Imported structure

This Part defines no agent-level operator and restates no law of the lower Parts. Table 1 lists every object it uses with the Part that owns it and the numbered statement in which that Part fixes it, and each is recalled with its attribution in the paragraphs that follow.

An interface is a pair (I, Σ) of pointed sets, I the per-tick input alphabet with idle symbol ε and Σ the per-tick output alphabet with idle symbol τ. An agent of sequential type $X \Rightarrow Y$ over (I, Σ) with failure alphabet E is a quadruple $(S, \mathrm{init}, \mathrm{step}, \mathrm{halt})$ with $\mathrm{init} : X \to S$, $\mathrm{step} : S \times I \to T(\Sigma \times S)$ and $\mathrm{halt} : S \to Y + E + 1$. A state is halted, failed or running according to the summand its image

lies in, and Part I requires that a state which is not running stutters: it emits τ and stays, whatever the input. Halting and failing are evaluated between ticks and consume no tick. The effect monad T is one of Id, the finite nonempty powerset monad $\mathcal{P}$, and the monad $\mathcal{D}$ of finitely supported rational probability distributions; all three are commutative and preserve weak pullbacks. Throughout this Part P denotes a transition kernel of a decision process and never the monad, which is written $\mathcal{P}$.

Part I factors a step into a policy and an update: $\mathrm{step}(s, i)$ is the computation that draws a pair (σ, k) from $\pi(s, i) \in T(\Sigma \times K)$ and returns $(\sigma, \delta(s, i, \sigma, k))$, where K records which resolution of the effect occurred. Every policy class in Section 3 is a constraint on π alone.

An environment for an agent over (I, Σ) is a co-agent E over the dual interface (Σ, I). Part I's closed loop $\mathrm{Run}(A, E)$ is the closed system over the trivial interface whose states are quadruples (s, q, σ, i) with (σ, i) a register initialized at (τ, ε) and the two components scheduled in lockstep: at each tick A consumes the I component of the register and E consumes the Σ component, and their outputs form the register for the next tick. The register is the only source of the delay discussed in Section 4. Proposition 3.8 of Part III proves that this concrete system is the delayed trace of the synchronous product, and Proposition 9.3 of Part I proves that for $\mathcal{D}$ with finite state spaces it is a Markov chain whose kernel is the product of the two step kernels.

Bisimilarity $\sim$ is defined by relation lifting of T on states related by the initializers and preserving halt; trace equivalence $\simeq_{tr}$ is equality of the trace semantics; refinement $\sqsubseteq$ is inclusion of stuttering-closed trace sets, and of their supports for $\mathcal{D}$. A predicate Φ on states is an invariant of an agent when it contains the image of the initializer and is closed under the transition (Definition 8.2 of Part I); Theorem 8.3 of Part I proves that invariants are closed under intersection and that the greatest invariant inside a predicate is the greatest fixed point of the predecessor operator.

From Part II this Part uses five constructions. Sequential composition $A\,;B$ runs A until it halts with a value y, then continues as B started at y, with no tick spent on the handoff and with failure propagating out of A without entering B. Guarded choice $A \oplus_b B$ selects a branch by evaluating a test b on the start value. Guarded recursion $\mu X.F(X)$ denotes the unique solution of $X = F(X)$ when F is guarded, meaning that every unfolding spends at least one tick before reaching the hole. The derived loop $\mathrm{while}_b(A)$ is $\mu X.((A\,;X) \oplus_b \mathsf{skip})$, and delay_d spends d ticks emitting τ and then halts with its start value. Theorem 5.1 of Part II proves that bisimilarity is a congruence for all of these, so every term below may be replaced by a bisimilar one inside any context.

From Part III this Part uses the approval gate $\mathrm{Gate}_H(A)$, in which A's proposed output is held while a human co-agent H is asked, A's state is frozen until an answer arrives, and the answer is one of ok, no and pending. From Part IV it uses replay determinism, Theorem 7.3 there, once, in Corollary 5.5.

Three symbols are this Part's own, and they are the ones the shared notation of the series assigns to it: π_θ for a parameterized stochastic policy, V for the value of a closed loop, and A^{Φ}_{π} for a shielded agent.

3 Policy classes

A policy in the sense of decision theory is not an extra piece of structure on an agent. It is the first component of the factorization that Part I already imposes on every step. Fixing a class of policies means constraining the codomain of π and the way π depends on the state.

Definition 3.1 (Policy classes)**.** Let $A = (S, \mathrm{init}, \mathrm{step}, \mathrm{halt})$ be an agent over (I, Σ) with $\Sigma = \mathsf{Act} \uplus \{\tau\}$ and $I = \mathsf{Obs} \uplus \{\varepsilon\}$, and let step be factored as (π, δ) in Part I's sense, with $\pi : S \times I \to T(\Sigma \times K)$. The policy component of A belongs to:

- Π_{det} when $T = \mathrm{Id}$ and K is a singleton, so that π is a function $S \times I \to \Sigma$;

- Π_{nd} when $T = \mathcal{P}$;
- Π_{stoch} when $T = \mathcal{D}$;
- Π_{plan} when $T = \mathrm{Id}$ and S is the set of residual plan terms of Definition 10.1, with π reading off the head action of the residual;
- Π_{ctrl} when $T = \mathrm{Id}$, Obs and Act are the state and input spaces of a linear plant, $S = \mathsf{Obs}$, and $\pi(x) = -Kx$ for a fixed real matrix K;
- Π_{LLM} when $T = \mathcal{D}$, S is a set of bounded contexts, and π is the composite of a parser with a language model kernel in the sense of Definition 5.1.

The six classes are not disjoint, and the relations among them are the relations among the three monads. A deterministic policy is a stochastic one whose distributions are Dirac and a nondeterministic one whose sets are singletons, because Id maps into $\mathcal{P}$ and $\mathcal{D}$ by their units. The map in the other direction, from stochastic to nondeterministic, is the support map, and the following lemma records that it is well behaved. The lemma is used again in Section 9, where it is what allows a statement about a stochastic policy to be derived from a statement about a nondeterministic one without any law ranging over a combined model of $\mathcal{P}$ and $\mathcal{D}$.

Lemma 3.2 (Support morphism)**.** *For every set X let* $\mathrm{supp} : \mathcal{D}(X) \to \mathcal{P}(X)$ *send a finitely supported rational distribution to the finite nonempty set of points it gives positive weight. Then* supp *is a morphism of monads:* $\mathrm{supp}(\eta_{\mathcal{D}}(x)) = \{x\}$ *and, for $\xi \in \mathcal{D}(X)$ and $g : X \to \mathcal{D}(Y)$,*

$$\mathrm{supp}(\xi \gg\!= g) \;=\; \bigcup_{x \in \mathrm{supp}(\xi)} \mathrm{supp}(g(x)).$$

Proof. The unit clause is immediate because a Dirac distribution gives positive weight to exactly one point. For the bind clause, $(\xi \gg\!= g)(y) = \sum_x \xi(x)\, g(x)(y)$, a finite sum of nonnegative rationals. Such a sum is positive if and only if some summand is positive, and $\xi(x)g(x)(y) > 0$ if and only if $x \in \mathrm{supp}(\xi)$ and $y \in \mathrm{supp}(g(x))$. Both sides therefore contain exactly the y for which such an x exists. Nonemptiness holds because ξ and each $g(x)$ are distributions and so have nonempty support. □

Definition 3.3 (Support abstraction of an agent)**.** For an agent A over $\mathcal{D}$, the agent $A^\sharp$ over $\mathcal{P}$ has the same state space, initializer and halt map, and $\mathrm{step}^\sharp(s, i) = \mathrm{supp}(\mathrm{step}(s, i))$.

Proposition 3.4 (Support abstraction preserves reachability)**.** *Let A be an agent over $\mathcal{D}$ and E an environment over $\mathcal{D}$. A state of* $\mathrm{Run}(A, E)$ *is reachable with positive probability if and only if it is reachable in* $\mathrm{Run}(A^\sharp, E^\sharp)$*.*

Proof. The n-tick transition of the closed loop is an n-fold bind of the two step maps and the register update, and the register update is a function, whose lifting to either monad is the graph map. By Lemma 3.2 applied n times, the support of the n-tick distribution of $\mathrm{Run}(A, E)$ is the set of n-tick successors of $\mathrm{Run}(A^\sharp, E^\sharp)$. Taking the union over n gives the claim. □

The last piece of vocabulary this section needs is a name for the routing construction that a policy performs when it selects among several agents rather than among several output symbols.

Definition 3.5 (Policy choice)**.** Let $b_1, \ldots, b_n$ be tests on a value type X that are pairwise disjoint and jointly exhaustive, computed from a deterministic policy $\pi : X \to \{1, \ldots, n\}$ by $b_i(x) = [\pi(x) = i]$. For agents $A_1, \ldots, A_n$ of type $X \Rightarrow Y$, write

$$\bigoplus\nolimits_{\pi}(A_1, \ldots, A_n) \coloneqq A_1 \oplus_{b_1} \left(A_2 \oplus_{b_2} (\cdots \oplus_{b_{n-1}} A_n)\right).$$

This is notation, not an operator. It abbreviates a nest of Part II's guarded choices, so every law it satisfies is an instance of a law Part II already proves, and none is restated here. Its only role below is to keep plan terms readable.

4 The decision epoch

Part I's closed loop passes each side's output to the other through a register that holds it for one tick. An agent acting at tick t therefore acts on information the environment emitted at tick $t-1$, and the environment applies that action at tick $t+1$. A construction that ignores this gets the timing wrong in a way that is not cosmetic: an action computed for one state is applied at another. The remedy used throughout this Part is to give the two sides matching phases, so that each decision occupies a fixed number of ticks and the action applied at a state is the action computed for that state.

Definition 4.1 (Epoch schedule)**.** Fix an interface with $\Sigma = \mathsf{Act} \uplus \{\tau\}$ and $I = \mathsf{Obs} \uplus \{\varepsilon\}$, where an observation carries a state and a reward slot, $\mathsf{Obs} = Q \times (\mathsf{Rew} \uplus \{\bot\})$ for a set Q of environment states and a set Rew of rational rewards.

An environment co-agent E over (Σ, I) is *sensor aligned* for a transition system $(Q, \mathsf{Act}, \mathrm{succ})$ with reward $R : Q \times \mathsf{Act} \to \mathsf{Rew}$ when $S_E = Q \times \{\mathrm{ann}, \mathrm{run}\}$, $\mathrm{init}(q_0) = (q_0, \mathrm{ann})$, and

$$\begin{aligned}
&\mathrm{step}\big((q, \mathrm{ann}), \sigma\big) = \eta\big((q, \bot),\ (q, \mathrm{run})\big) \quad \text{for every } \sigma,\\
&\mathrm{step}\big((q, \mathrm{run}), \tau\big) = \eta\big(\varepsilon,\ (q, \mathrm{run})\big),\\
&\mathrm{step}\big((q, \mathrm{run}), a\big) = \big(q' \leftarrow \mathrm{succ}(q, a);\ \eta\big((q', R(q, a)),\ (q', \mathrm{run})\big)\big) \quad \text{for } a \in \mathsf{Act},
\end{aligned}$$

where $\mathrm{succ}(q, a) \in T(Q)$ is the environment's move. A sensor-aligned environment is *idle stationary*: an idle output leaves its state unchanged and produces no observation.

A leaf sense of type $W \Rightarrow W'$ is a *sensing leaf* when, on input ε, it emits τ and stays, and on an input in Obs it emits τ and halts with a value computed from its start value, the observation, and one draw from T. An agent emit of type $W' \Rightarrow W$ is an *emitter* when it spends exactly one tick, emits an element of Act determined by its start value, and halts. A guarded choice of two emitters is an emitter, because the test consumes no tick.

An agent A over (I, Σ) is *actuator aligned* when it is a guarded recursion each of whose unfoldings is a sensing leaf followed by either an emitter and a recursive occurrence, or an agent that halts at once. The basic case is

$$A \;=\; \mu X.\big((\mathsf{sense}\,;\mathsf{emit})\,;X\big),$$

and the options, shields and plans of the later sections are the remaining cases.

The definition puts every effect in a leaf. The sampling of an action, the drawing of a termination coin and the update of a belief all happen inside sense, and emit is a deterministic one-tick emitter. This keeps the guarded choices of the later sections Boolean, which is what Part II's tests require, and it means that the only place a monad appears is where Part I already puts it.

Lemma 4.2 (Epoch adequacy)**.** *Let A be actuator aligned and E sensor aligned for $(Q, \mathsf{Act}, \mathrm{succ}, R)$ over the same monad T, and consider $\mathrm{Run}(A, E)$ started at environment state q_0. Then, until A halts or fails:*

1. *the run decomposes into epochs, the k-th occupying ticks $3k+1$, $3k+2$ and $3k+3$; in the first the composite is inside sense and receives ε, in the second it is inside sense and receives an observation, and in the third it is inside emit and emits an action $a_k \in \mathsf{Act}$;*

2. *the environment's Q component equals q_k throughout ticks $3k+2$ and $3k+3$, and q_{k+1} is drawn from* $\mathrm{succ}(q_k, a_k)$ *at tick $3k+4$;*

3. *the observation read at tick $3k+2$ is $(q_k, R(q_{k-1}, a_{k-1}))$ for $k \geq 1$ and $(q_0, \bot)$ for $k = 0$, so the sequence of rewards appearing in the run is exactly $R(q_0, a_0), R(q_1, a_1), \ldots$;*

4. *for $T = \mathcal{D}$ with Q,* Act *and the leaves' state spaces finite, the epoch process $(q_k, a_k)_{k \geq 0}$ is a Markov chain whose action is drawn by* sense *from q_k and whose state moves by* succ.

Proof. Write σ_t and i_t for the outputs of A and E at tick t, with $\sigma_0 = \tau$ and $i_0 = \varepsilon$ the register initialization of Part I's closed loop; at tick t the agent consumes i_{t-1} and the environment consumes σ_{t-1}.

At tick 1 the agent is in sense and consumes $i_0 = \varepsilon$, so it emits τ and stays; the environment is in ann and consumes $\sigma_0 = \tau$, so it emits $(q_0, \bot)$ and moves to (q_0, run). At tick 2 the agent consumes $i_1 = (q_0, \bot)$, emits τ and halts, handing on to emit without a tick; the environment consumes $\sigma_1 = \tau$ and stays. At tick 3 the agent emits a_0 and halts, handing on to the next unfolding of the recursion; the environment consumes $\sigma_2 = \tau$ and stays. At tick 4 the agent is again in sense and consumes $i_3 = \varepsilon$, so it stays; the environment consumes $\sigma_3 = a_0$, draws q_1 from $\mathrm{succ}(q_0, a_0)$ and emits $(q_1, R(q_0, a_0))$. At tick 5 the agent consumes that observation and halts. This is the configuration of tick 2 with k increased by one, so the pattern repeats, which gives (1), (2) and (3). The handoffs consume no tick by Part I's convention that halting is evaluated between ticks, and the recursion is guarded because emit spends a tick, so Theorem 7.1 of Part II makes the displayed term denote this run and no other.

For (4), the closed loop has finitely many states, so by Proposition 9.3 of Part I it is a Markov chain on the product of the two state spaces with the register. The epoch process is the image of that chain under the map sending the configuration at tick $3k+3$ to (q_k, a_k), and by (1) and (2) the ticks $3k+4$ to $3k+6$ depend on the past only through that configuration. Hence the epoch process is Markov, with the action drawn by sense from q_k and the state moved by succ. □

Remark 4.3 (Cost of the register). Three ticks per decision is the price of Part I's register together with the requirement that sense and emit be separate leaves. Fusing them into a single leaf that reads an observation and emits an action in the same tick gives a two-tick epoch and the same epoch process; the split is kept because Sections 7 and 9 need a point between reading and acting at which a Boolean test can be evaluated. Nothing below depends on the number three: every statement is about the epoch process, which Lemma 4.2 identifies for either arrangement.

5 Language model loops

A tool-calling loop over a language model is an agent as soon as one is explicit about what is bounded. The model itself is a map from a context to a distribution over token sequences. Two boundedness assumptions turn that map into a step function of a finite-state agent over $\mathcal{D}$, and a third makes the distributions rational.

Definition 5.1 (Language model policy)**.** Fix a finite token vocabulary V, a context bound L and a generation bound N. Put $\mathsf{Ctx} \coloneqq V^{\leq L}$ and $\mathsf{Tok} \coloneqq V^{\leq N}$, both finite sets. A *language model policy* is a map $\pi_\theta : \mathsf{Ctx} \to \mathcal{D}(\mathsf{Tok})$ together with a parser $\mathrm{parse} : \mathsf{Tok} \to \Sigma$ and a truncation $\mathrm{trunc} : V^* \to \mathsf{Ctx}$ keeping the last L tokens. The *bounded-generation abstraction* is the conjunction of three assumptions: the context is truncated to L tokens before every call; generation stops after at most N tokens; and the weights of $\pi_\theta(c)$ are rational for every c.

Remark 5.2 (What the abstraction assumes). Rationality is the weakest of the three assumptions, since a sampler that draws from a rounded softmax draws from a distribution whose weights are dyadic rationals, and dyadic rationals are rational. The substantive assumptions are the two bounds, which make Ctx finite and so make π_θ a kernel between finite sets. A serving stack whose output distribution depends on the batch a request is placed in does not present a fixed kernel at all, and no statement of this section applies to it. The abstraction is about inference; it says nothing about a change of θ.

The loop that ReAct describes interleaves a reasoning step with a tool call and stops when the model emits a finishing action. Both steps are epoch-aligned agents in the sense of Definition 4.1, and the loop is Part II's derived while.

Definition 5.3 (ReAct term)**.** Let Think and Act be the agents of type $\mathsf{Ctx} \Rightarrow \mathsf{Ctx}$ defined as follows. Think starts at a context c; its sensing leaf waits for the tool observation o, draws a thought t from $\pi_\theta(\mathrm{trunc}(c \cdot o \cdot \text{“}\mathsf{Thought}\text{:”}))$ and halts with $\mathrm{trunc}(c \cdot o \cdot t)$; its emitter emits the thought as an output in a reserved subset of Σ. Act starts at a context c; its sensing leaf draws an action string u from $\pi_\theta(\mathrm{trunc}(c \cdot \text{“}\mathsf{Action}\text{:”}))$ and halts with $\mathrm{trunc}(c \cdot u)$; its emitter emits $\mathrm{parse}(u) \in \mathsf{Act}$. Let $\mathrm{done} : \mathsf{Ctx} \to 2$ test whether the last parsed action was the finishing action. The *ReAct term* is

$$\mathrm{ReAct} \coloneqq \mu X.(((\mathsf{Think} \,;\, \mathsf{Act}) \,;\, X) \oplus_{\mathrm{done}} \mathsf{skip}) \;=\; \mathrm{while}_{\neg \mathrm{done}}(\mathsf{Think} \,;\, \mathsf{Act}).$$

Proposition 5.4 (A bounded tool loop is an agent)**.** *Under the bounded-generation abstraction,* Think*,* Act *and* ReAct *are agents over* $\mathcal{D}$ *in Part I's sense, with finite state spaces. The recursion in Definition 5.3 is guarded, so it has a unique solution up to bisimilarity, and that solution is bisimilar to the hand-written loop agent* L *whose state space is* $\mathsf{Ctx} \times \{\mathit{think}, \mathit{emit\text{-}thought}, \mathit{act}, \mathit{emit\text{-}action}\}$*, whose step performs the corresponding draw or emission, and which halts with the current context in the state* (c, think) *when* $\mathrm{done}(c)$*.*

Proof. Each leaf has a finite state space because Ctx and Tok are finite, and each step returns a finitely supported rational distribution because $\pi_\theta(c)$ does and the remaining data are functions of the draw. The halt map sends the halting states of each leaf into the value type and every other state to running, and non-running states stutter because the leaves are defined to stutter once halted. So each leaf is an agent, and Think, Act and the composite are agents because Part II's operators preserve agenthood.

Guardedness: every unfolding of the recursion passes through the emitter of Act, which spends a tick before the recursive occurrence is reached. Theorem 7.1 of Part II therefore gives a solution unique up to bisimilarity.

For the bisimulation, relate an unfolding term to the state of L that carries the same context and the same phase: relate $\mathsf{Think} \,;\, \mathsf{Act} \,;\, \mathrm{ReAct}$ started at c to (c, think), the residual after Think's sensing leaf halts to $(c', \mathrm{emit\text{-}thought})$, and so on, and relate skip started at c to the halted state carrying c. Each pair has matching halt values, and for each input the two step distributions are equal as elements of $\mathcal{D}(\Sigma \times \cdot)$ up to the relation, because both are the image of the same draw from π_θ under the same function. The relation is therefore a bisimulation containing the pair of initial states, so the two agents are bisimilar. □

Corollary 5.5 (Logged loops replay)**.** *Let* $\ell = (i_1, r_1) \cdots (i_N, r_N)$ *be a log of inputs and choice resolutions that is compatible with* ReAct *from the start context, in the sense of Part IV. Then the replayed agent is an agent over* Id *whose state trajectory over the first* N *ticks is the logged trajectory and whose erased output word is the logged output word.*

Proof. Theorem 7.3 of Part IV applies to an agent over any of the three monads together with a compatible log of inputs and choice resolutions. Proposition 5.4 supplies the hypothesis that ReAct is such an agent, and its choice resolutions are the draws from π_θ, which the log records. Clause (ii) of that theorem adds that replaying the same log against a second policy $\pi_{\theta'}$ of the same shape either reproduces the logged output word or fails with the reason recording nondeterminism, which is how a change of model is detected rather than silently absorbed. □

Remark 5.6 (Temperature zero). Greedy decoding makes every $\pi_\theta(c)$ a Dirac distribution, so the policy lies in the image of Id under the unit and the loop is deterministic. This is a statement about the abstraction, not about an implementation: a decoder whose arithmetic depends on batch composition can return different argmaxes for the same context, and then the observed behavior is not a function of the context at all.

6 Markov environments and closed-loop value

A Markov decision process supplies an environment, not a new kind of object. The ruling of this series is that Part I owns the environment and this Part contributes an instance.

Definition 6.1 (Markov decision environment)**.** Let $M = (Q, \mathsf{Act}, P, R, \gamma)$ be a finite Markov decision process with rational transition kernel $P(\cdot \mid q, a) \in \mathcal{D}(Q)$, rational reward $R : Q \times \mathsf{Act} \to \mathsf{Rew}$ and rational discount $\gamma \in [0, 1)$. The *Markov decision environment* E_M is the sensor-aligned environment of Definition 4.1 over $T = \mathcal{D}$ with $\mathrm{succ}(q, a) = P(\cdot \mid q, a)$ and reward R. This is an instance of Part I's environment, not a new definition: it is the co-agent obtained from M by fixing the phase structure and the reward slot.

Definition 6.2 (Closed-loop value)**.** Let A be actuator aligned and E sensor aligned over $\mathcal{D}$. The *reward read-out* of the run is the sequence $(r_k)_{k \geq 0}$ of the non-$\bot$ reward slots of the observations, in tick order. For $\gamma \in [0, 1)$ the *discounted closed-loop value* of $\mathrm{Run}(A, E)$ started at q is

$$V_\gamma(\mathrm{Run}(A, E))(q) \coloneqq \mathbb{E}\Big[\sum_{k \geq 0} \gamma^k r_k\Big],$$

where the expectation is taken over the distribution the closed loop assigns to runs, and a run in which the read-out is finite contributes only its finitely many terms.

The series converges absolutely whenever the rewards are bounded, which they are for a finite M: $\sum_k \gamma^k |r_k| \leq R_{\max}/(1 - \gamma)$ with $R_{\max} = \max_{q,a} |R(q, a)|$. The value is therefore well defined and expectation and summation may be exchanged.

Definition 6.3 (Memoryless stochastic policy agent)**.** For $\pi : Q \to \mathcal{D}(\mathsf{Act})$ with rational weights, let $A_\pi \coloneqq \mu X.((\mathsf{sense}_\pi \,;\, \mathsf{emit}) \,;\, X)$, where sense_π is the sensing leaf of type $1 \Rightarrow Q \times \mathsf{Act}$ that, on reading (q, r), draws a from $\pi(q)$ and halts with (q, a), and emit is the emitter of type $Q \times \mathsf{Act} \Rightarrow 1$ that emits a and halts.

Theorem 6.4 (Closed-loop value is the policy value)**.** *Let M be a finite Markov decision process with rational data and $\gamma \in [0, 1)$, and let $\pi : Q \to \mathcal{D}(\mathsf{Act})$ be a memoryless stochastic policy with rational weights. Then, for every $q_0 \in Q$,*

$$V_\gamma(\mathrm{Run}(A_\pi, E_M))(q_0) \;=\; V^\pi(q_0),$$

where V^π is the unique solution of the policy evaluation equation $V = R_\pi + \gamma P_\pi V$ with $R_\pi(q) = \sum_a \pi(a \mid q) R(q, a)$ and $P_\pi(q' \mid q) = \sum_a \pi(a \mid q) P(q' \mid q, a)$.

Proof. The agent A_π is actuator aligned and E_M is sensor aligned, and both state spaces are finite, so Lemma 4.2 applies. By clauses (1) to (3) the reward read-out of the run is exactly $(R(q_k, a_k))_{k\geq 0}$, and by clause (4) the epoch process (q_k, a_k) is the Markov chain in which a_k is drawn from $\pi(q_k)$ and q_{k+1} from $P(\cdot \mid q_k, a_k)$. This is the chain of M under π, so the marginal law of q_k is P_π^k applied to the point mass at q_0.

Rewards are bounded by $R_{\max}$ and $\gamma < 1$, so $\sum_k \gamma^k \mathbb{E}|r_k| < \infty$ and Fubini's theorem permits the exchange

$$V_\gamma(\mathrm{Run}(A_\pi, E_M))(q_0) \;=\; \sum_{k\geq 0} \gamma^k \, \mathbb{E}[R(q_k, a_k)] \;=\; \sum_{k\geq 0} \gamma^k \, (P_\pi^k R_\pi)(q_0),$$

where the second equality uses $\mathbb{E}[R(q_k, a_k) \mid q_k] = R_\pi(q_k)$, which holds because a_k is drawn from $\pi(q_k)$ independently of the past given q_k. The matrix $I - \gamma P_\pi$ is invertible because P_π is stochastic and hence has spectral radius one, so γP_π has spectral radius at most $\gamma < 1$; the Neumann series $\sum_k \gamma^k P_\pi^k$ converges to $(I - \gamma P_\pi)^{-1}$. The displayed sum is therefore $((I - \gamma P_\pi)^{-1} R_\pi)(q_0)$, which is the unique solution of $V = R_\pi + \gamma P_\pi V$ evaluated at q_0. That this solution is the discounted value of π in M is the policy evaluation equation of discounted dynamic programming, Puterman (1994), Chapter 6. □

Corollary 6.5 (Evaluation and improvement act on agents)**.** *Fix M and γ. Policy evaluation sends the agent A_π to the function V^π by Theorem 6.4. Policy improvement sends A_π to $A_{\pi'}$ with $\pi'(q)$ any distribution supported on $\arg\max_a \{R(q, a) + \gamma \sum_{q'} P(q' \mid q, a) V^\pi(q')\}$, and then $V_\gamma(\mathrm{Run}(A_{\pi'}, E_M)) \geq V_\gamma(\mathrm{Run}(A_\pi, E_M))$ pointwise. Iterating the two maps from any A_π reaches, in finitely many steps, an agent whose closed-loop value is the optimal value of M.*

Proof. Both maps are well defined on agents of the form A_π because Definition 6.3 makes A_π a function of π and Theorem 6.4 makes the closed-loop value a function of π. The inequality and the finite termination are the policy improvement theorem and the convergence of policy iteration for finite discounted models, Puterman (1994), Chapter 6, applied to π and π'; Theorem 6.4 transports them from values of policies to values of closed loops. □

Example 6.6 (A two-state chain)**.** Let $Q = \{q_0, q_1\}$, $\mathsf{Act} = \{a, b\}$ and $\gamma = 1/2$. Let $P(q_1 \mid q_0, a) = 1$, $P(q_0 \mid q_0, b) = 1$ and $P(q_0 \mid q_1, \cdot) = 1$; let $R(q_0, a) = 1$ and $R = 0$ elsewhere. Take the deterministic policy $\pi(q_0) = \pi(q_1) = a$. The epoch process alternates $q_0, q_1, q_0, \ldots$ and the reward read-out is $1, 0, 1, 0, \ldots$, so the closed-loop value at q_0 is $\sum_{j\geq 0} \gamma^{2j} = 1/(1-\gamma^2) = 4/3$ and at q_1 it is $\gamma \cdot 4/3 = 2/3$. The evaluation equations give the same pair: $V(q_0) = 1 + \frac{1}{2} V(q_1)$ and $V(q_1) = \frac{1}{2} V(q_0)$ have the unique solution $(4/3, 2/3)$. In ticks, the first action is emitted at tick 3 and the first reward appears in the observation the environment emits at tick 4.

7 Options

An option is a policy with a stopping rule. Written in the algebra it is a guarded recursion whose body is one epoch and whose exit test reads a coin drawn while sensing. The value equation that the options framework proves about it turns out to be a statement about sequential composition, and that is the content of this section.

Definition 7.1 (Option term)**.** Let M be a finite Markov decision process and let $o = (I_o, \pi_o, \beta_o)$ consist of an initiation set $I_o \subseteq Q$, an internal policy $\pi_o : Q \to \mathcal{D}(\mathsf{Act})$ and a rational termination probability $\beta_o : Q \to [0, 1] \cap \mathbb{Q}$. Let sense_o be the sensing leaf of type $1 \Rightarrow Q \times \mathsf{Act} \times 2$ that, on reading (q, r), draws a from $\pi_o(q)$ and c from the Bernoulli distribution with parameter $\beta_o(q)$, and halts with (q, a, c). Let emit be the emitter of type $Q \times \mathsf{Act} \times 2 \Rightarrow 1$ that emits a, and let ret be the

agent of type $Q \times \mathsf{Act} \times 2 \Rightarrow Q$ that halts at once with q. With the test $\mathrm{cont}(q, a, c) = [c = 0]$, the *option term* is

$$A_o \coloneqq \mu X.\Big(\mathsf{sense}_o \,;\, ((\mathsf{emit} \,;\, X) \oplus_{\mathrm{cont}} \mathsf{ret})\Big) \;:\; 1 \Rightarrow Q.$$

The term is guarded, because every unfolding passes through emit, so Theorem 7.1 of Part II gives it a meaning. Its behavior is the behavior the options framework prescribes: on entering a state q the option terminates with probability $\beta_o(q)$ before acting there, and otherwise takes an action drawn from $\pi_o(q)$. When β_o takes only the values 0 and 1 the coin is redundant and the test can be taken directly on q, in which case the term is Part II's $\mathrm{while}_{\neg\beta_o}$ applied to one epoch. The general case keeps the randomness inside the leaf so that the guard stays Boolean, which is what Part II's tests require.

Definition 7.2 (Epoch-aligned halting)**.** An actuator-aligned agent A *halts at epoch boundaries* when every halting state of A is reached between the last tick of an epoch and the first tick of the next, so that the number τ of completed epochs is well defined and the environment is in state q_τ when A halts.

Option terms halt at epoch boundaries: ret halts at once, immediately after the sensing tick of the epoch it declines to complete, and sense has not yet caused the environment to move. The initiation set plays no part in the value equation and appears only as a precondition on where A_o may be started.

Theorem 7.3 (Value composes along sequential composition)**.** *Let M be a finite Markov decision process with rational data and $\gamma \in [0, 1)$, and let $A : 1 \Rightarrow Q$ and $B : Q \Rightarrow Y$ be actuator-aligned agents over $\mathcal{D}$ with finite state spaces such that A halts at epoch boundaries with halting value q_τ, where the halting epoch τ satisfies $\mathbb{E}[\tau] < \infty$. Then for every $q \in Q$,*

$$V_\gamma(\mathrm{Run}(A \,;\, B, E_M))(q) \;=\; \mathbb{E}_q\Big[\sum_{k<\tau} \gamma^k r_k \;+\; \gamma^\tau\, V_\gamma(\mathrm{Run}(B, E_M))(q_\tau)\Big],$$

with (r_k) the reward read-out of the run.

Proof. Write Z_k for the state of the closed loop at the start of epoch k, a pair consisting of the current state of the composite agent and the environment state q_k. All state spaces are finite, so by Lemma 4.2 clause (4) and Proposition 9.3 of Part I, $(Z_k)_{k\geq 0}$ is a Markov chain.

The handoff in $A \,;\, B$ consumes no tick and replaces the state of A by $\mathrm{init}_B(q_\tau)$, leaving the environment untouched. By Definition 7.2 the handoff falls between epochs, so the epochs of the run split into the first τ, produced by A, and the rest, produced by B started at q_τ. Hence the reward read-out satisfies

$$\sum_{k\geq 0} \gamma^k r_k \;=\; \sum_{k<\tau} \gamma^k r_k \;+\; \gamma^\tau \sum_{j\geq 0} \gamma^j r_{\tau+j}$$

pointwise on runs, both sides being absolutely convergent because $|r_k| \leq R_{\max}$ and $\gamma < 1$.

The event $\{\tau \leq k\}$ is determined by $Z_0, \ldots, Z_k$, because whether A has halted is a property of the composite agent's state. So τ is a stopping time of the filtration generated by (Z_k), and $\mathbb{E}[\tau] < \infty$ gives $\tau < \infty$ almost surely. The strong Markov property of a finite-state Markov chain at τ says that, conditionally on the history up to τ, the process $(Z_{\tau+j})_{j\geq 0}$ is the chain started at $Z_\tau = (\mathrm{init}_B(q_\tau), q_\tau)$. That chain is the epoch chain of $\mathrm{Run}(B, E_M)$ started at q_τ, so

$$\mathbb{E}\Big[\sum_{j\geq 0} \gamma^j r_{\tau+j} \;\Big|\; \mathcal{F}_\tau\Big] \;=\; V_\gamma(\mathrm{Run}(B, E_M))(q_\tau).$$

Taking expectations in the pointwise identity, using dominated convergence with the summable bound $R_{\max}\gamma^k$ to exchange expectation and summation, and using the tower property on the second piece gives the statement. □

Remark 7.4 (Relation to the semi-Markov equation). With A an option term and B the agent that continues under a policy over options, the displayed identity is the Bellman equation for options, equation (8) of Section 3 of Sutton, Precup and Singh (1999), up to the index convention that starts the reward stream at $k = 0$ rather than at $t+1$. Their Theorem 1 of the same section, that an MDP with a fixed set of options is a semi-Markov decision process, is the statement that the epoch chain restricted to option boundaries is Markov with the transit times supplied by τ; that is the strong Markov step of the proof above. The hypothesis $\mathbb{E}[\tau] < \infty$ is what makes γ^τ and the truncated sum integrable without further conditions, and it is not implied by the option's definition: an option whose termination probability vanishes on a recurrent set never halts.

Example 7.5 (An option in a corridor)**.** Let $Q = \{0, 1, 2\}$, $\mathsf{Act} = \{\mathrm{R}\}$, $P(\min(q+1, 2) \mid q, \mathrm{R}) = 1$, $R(q, \mathrm{R}) = -1$ for $q < 2$ and $R(2, \mathrm{R}) = 0$, and $\gamma = 1/2$. Let o have $\pi_o = \mathrm{R}$ and $\beta_o(q) = [q = 2]$. Started at 0, the option acts at 0 and at 1 and terminates on entering 2, so $\tau = 2$, the reward read-out of its epochs is $-1, -1$, and $\sum_{k<2} \gamma^k r_k = -1 - \frac{1}{2} = -\frac{3}{2}$. Theorem 7.3 then gives $V_\gamma(\mathrm{Run}(A_o \,;\, B, E_M))(0) = -\frac{3}{2} + \frac{1}{4} V_\gamma(\mathrm{Run}(B, E_M))(2)$ for every continuation B, and the same number results from summing the reward read-out of the composite run directly.

8 Belief agents

Partial observability changes what the agent may condition on, not what an agent is. A finite partially observable process induces an agent whose state is a distribution over the hidden states and whose update is Bayes' rule. Two features of Part I's signature earn their place here: the state space of an agent is an arbitrary set, so a countably infinite set of reachable beliefs is allowed, and failure is a halt outcome, which is where an impossible observation goes.

Definition 8.1 (Belief agent)**.** Let $N = (Q, \mathsf{Act}, \Omega, P, O, R, \gamma)$ be a finite partially observable Markov decision process with rational transition kernel $P(q' \mid q, a)$, rational observation kernel $O(\omega \mid q', a)$, rational reward R, rational discount $\gamma \in [0, 1)$ and rational initial belief $b_0 \in \mathcal{D}(Q)$. The *state estimator* is

$$\mathrm{SE}(b, a, \omega)(q') \;=\; \frac{O(\omega \mid q', a) \sum_q P(q' \mid q, a)\, b(q)}{Z(b, a, \omega)},$$
$$Z(b, a, \omega) \;=\; \sum_{q'} O(\omega \mid q', a) \sum_q P(q' \mid q, a)\, b(q),$$

which is Bayes' rule in the form of Section 3.3 of Kaelbling, Littman and Cassandra (1998). Fix a belief policy $\hat{\pi} : \mathcal{D}(Q) \to \mathcal{D}(\mathsf{Act})$ with rational weights. The *belief agent* $\mathrm{Bel}_{\hat{\pi}} : \mathcal{D}(Q) \times \mathsf{Act} \Rightarrow Y$ is the actuator-aligned term $\mu X.((\mathsf{sense}^{\mathrm{bel}} \,;\, \mathsf{emit}) \,;\, X)$ whose sensing leaf, started at (b, a) and reading the observation ω, halts with $(\mathrm{SE}(b, a, \omega), a')$ where a' is drawn from $\hat{\pi}(\mathrm{SE}(b, a, \omega))$, and fails with the reason $\mathsf{impossible\text{-}observation}$ when $Z(b, a, \omega) = 0$; the initial announcement carries the distinguished symbol start and leaves the belief unchanged. The emitter emits a'.

Proposition 8.2 (The belief agent is an agent)**.** *$\mathrm{Bel}_{\hat{\pi}}$ is an agent over $\mathcal{D}$ in Part I's sense. Its reachable state space is countable, and every reachable belief has rational weights. In the closed loop with the environment of Definition 8.3 the failure state is unreachable.*

Proof. The estimator maps rationals to rationals, since P, O and b are rational and the quotient of rationals is rational, so from a rational b_0 every reachable belief is rational. The set of reachable beliefs is the union over k of the images of SE iterated k times on the finite set $\mathsf{Act} \times \Omega$, hence a countable union of finite sets. Each step returns a finitely supported rational distribution, and non-running states stutter by the construction of the leaves, so the quadruple is an agent. The failure branch is entered only when $Z(b, a, \omega) = 0$, and $Z(b, a, \omega)$ is the probability the environment assigns to emitting ω after a from belief b; an observation of probability zero is not emitted, so no run of the closed loop reaches the failure state. □

Definition 8.3 (Partially observable environment)**.** E_N is the sensor-aligned environment over $\mathcal{D}$ whose state is the hidden state q, whose move on action a draws q' from $P(\cdot \mid q, a)$ and emits the observation ω drawn from $O(\cdot \mid q', a)$ together with the reward $R(q, a)$, and whose initial announcement carries start with reward slot $\bot$.

Lemma 8.4 (The carried belief is the posterior)**.** *In* $\mathrm{Run}(\mathrm{Bel}_{\hat\pi}, E_N)$ *started at* (b_0, q_0) *with* q_0 *drawn from* b_0*, let* b_k *be the belief carried at the start of epoch* k *and let* $h_k = (a_0, \omega_1, \ldots, a_{k-1}, \omega_k)$ *be the sequence of actions emitted and observations read. Then for every history* h_k *of positive probability and every* $q \in Q$*,*

$$b_k(q) \;=\; \Pr\left[q_k = q \mid h_k\right].$$

Proof. Induction on k. For $k = 0$ the belief is b_0 and the hidden state is drawn from b_0, and the initial announcement carries no observation, so the two sides agree. Assume the claim for k. Conditioning on h_k and on the action a_k, which is drawn from $\hat\pi(b_k)$ and hence is independent of q_k given h_k, the joint law of (q_{k+1}, ω_{k+1}) has density $\sum_q b_k(q) P(q' \mid q, a_k) O(\omega \mid q', a_k)$ at (q', ω). Dividing by the marginal in ω, which is $Z(b_k, a_k, \omega)$, gives $\Pr[q_{k+1} = q' \mid h_k, a_k, \omega_{k+1} = \omega] = \mathrm{SE}(b_k, a_k, \omega)(q')$. The belief carried at the start of epoch $k + 1$ is exactly $\mathrm{SE}(b_k, a_k, \omega_{k+1})$, which proves the claim for $k + 1$. This is the sufficiency of the belief state recorded in Section 3.2 of Kaelbling, Littman and Cassandra (1998) and, for the general partially observed control problem, by Astrom (1965). □

Lemma 8.5 (The belief process is Markov)**.** *Under the hypotheses of Lemma 8.4, the process* $(b_k)_{k \geq 0}$ *is a Markov chain on the countable set of reachable beliefs, with kernel*

$$\bar P(b' \mid b, a) \;=\; \sum_{\omega \,:\, \mathrm{SE}(b,a,\omega) = b'} Z(b, a, \omega),$$

and the action a_k *is drawn from* $\hat\pi(b_k)$*.*

Proof. By Lemma 4.2 the epoch process of the closed loop is determined by the state of the composite at the start of an epoch, which consists of the carried pair (b_k, a_{k-1}) and the hidden state q_k. The distribution of ω_{k+1} given (b_k, a_k) is $Z(b_k, a_k, \cdot)$ by the computation in Lemma 8.4, and it depends on the past only through b_k and a_k. Since $b_{k+1} = \mathrm{SE}(b_k, a_k, \omega_{k+1})$ and a_k is drawn from $\hat\pi(b_k)$, the law of b_{k+1} given the past depends only on b_k, with the displayed kernel. The state space is countable by Proposition 8.2; countability is all that is used, and the finiteness hypothesis of Proposition 9.3 of Part I is not invoked. □

Theorem 8.6 (Belief agents carry the value)**.** *Let* N *be a finite partially observable Markov decision process with rational data and* $\gamma \in [0, 1)$*, and let* $\hat\pi$ *be a rational belief policy. Then*

$$V_\gamma(\mathrm{Run}(\mathrm{Bel}_{\hat\pi}, E_N))(b_0) \;=\; \bar V^{\hat\pi}(b_0),$$

where $\bar V^{\hat\pi}$ *is the value of* $\hat\pi$ *in the belief process with state space the reachable beliefs, kernel* $\bar P$ *of Lemma 8.5 and reward* $\bar R(b, a) = \sum_q b(q) R(q, a)$*.*

Proof. The reward read-out of the run is $(R(q_k, a_k))_{k\geq 0}$ by Lemma 4.2 clause (3), applied to E_N, whose reward slot carries $R(q_k, a_k)$. Rewards are bounded because Q and Act are finite, so the value is an absolutely convergent series and expectation may be exchanged with summation. Conditioning on the history up to epoch k and using Lemma 8.4,

$$\mathbb{E}\big[R(q_k, a_k) \mid h_k, a_k\big] \;=\; \sum_q \Pr[q_k = q \mid h_k]\, R(q, a_k) \;=\; \sum_q b_k(q) R(q, a_k) \;=\; \bar{R}(b_k, a_k),$$

where the first equality uses that a_k is drawn from $\hat{\pi}(b_k)$ and is therefore conditionally independent of q_k given h_k. Hence $\mathbb{E}[R(q_k, a_k)] = \mathbb{E}[\bar{R}(b_k, a_k)]$ for every k, and

$$V_\gamma\big(\mathrm{Run}(\mathrm{Bel}_{\hat{\pi}}, E_N)\big)(b_0) \;=\; \sum_{k\geq 0} \gamma^k\, \mathbb{E}\big[\bar{R}(b_k, a_k)\big].$$

By Lemma 8.5 the right-hand side is the discounted value of $\hat{\pi}$ in the belief process, which is $\bar{V}^{\hat{\pi}}(b_0)$. The identification of that process as the fully observed decision process equivalent to N is the belief reduction of Section 3.4 of Kaelbling, Littman and Cassandra (1998). □

Example 8.7 (Two states and a noisy sensor)**.** Let $Q = \{q_1, q_2\}$, let the only action leave the hidden state fixed, let $\Omega = \{\omega_1, \omega_2\}$ with $O(\omega_1 \mid q_1) = O(\omega_2 \mid q_2) = 3/4$, and let $b_0 = (\frac{1}{2}, \frac{1}{2})$. Reading ω_1 gives $Z = \frac{1}{2}$ and the posterior $(\frac{3}{4}, \frac{1}{4})$. Reading ω_1 again gives $Z = \frac{5}{8}$ and the posterior $(\frac{9}{10}, \frac{1}{10})$. Every quantity stays rational, which is what Definition 8.1 needs for the belief agent to live over $\mathcal{D}$ as Part I fixes it; a sensor with irrational accuracy leaves the fragment and its belief agent is not an object of this algebra.

9 Shields and controllable invariants

A shield watches what a policy is about to do and replaces the action when it would break a safety requirement. In the algebra this is one guarded choice inserted between sensing and emitting, and the safety requirement is an invariant in Part I's sense. The condition under which the construction works has a name in the reactive synthesis literature and is the only hypothesis the theorem needs.

Definition 9.1 (Controllable invariant)**.** Let $(Q, \mathsf{Act}, \mathrm{succ})$ be a transition system with $\mathrm{succ}(q, a)$ a nonempty finite subset of Q, and let $\Phi \subseteq Q$. The *controllable predecessor* of $\Psi \subseteq Q$ is

$$\mathrm{CPre}(\Psi) \;=\; \{\, q \in Q \;:\; \exists\, a \in \mathsf{Act}.\ \mathrm{succ}(q, a) \subseteq \Psi \,\}.$$

Φ is *controllable* when $\Phi \subseteq \mathrm{CPre}(\Phi)$. The *safe action set* at $q \in \Phi$ is $\mathrm{Safe}(q) = \{a \in \mathsf{Act} : \mathrm{succ}(q, a) \subseteq \Phi\}$, which is nonempty exactly when $q \in \mathrm{CPre}(\Phi)$. A *fallback* is a map $f : \Phi \to \mathsf{Act}$ with $f(q) \in \mathrm{Safe}(q)$.

The quantification in CPre ranges over Act and not over Σ. Excluding the idle symbol is what makes the condition substantive: an environment that does not move when the agent idles would make every predicate controllable by idling forever, and the resulting shield would be safe and useless. Section 9 returns to that degenerate reading in Proposition 9.7, where it is exactly what a human approval gate does.

Definition 9.2 (Shielded agent)**.** Let π be a policy, Φ a controllable predicate with fallback f, and $A_\pi = \mu X.((\mathsf{sense}_\pi \,;\, \mathsf{emit}) \,;\, X)$ the unshielded loop of Definition 6.3. Let emit_f be the emitter of type $Q \times \mathsf{Act} \Rightarrow 1$ that emits $f(q)$ rather than a, and let $\mathrm{safe}(q, a) = [\, a \in \mathrm{Safe}(q) \,]$, a test on the value handed on by sense_π. The *shielded agent* is

$$A_\pi^\Phi \coloneqq \mu X.\Big(\big(\mathsf{sense}_\pi \,;\, (\mathsf{emit} \oplus_{\mathrm{safe}} \mathsf{emit}_f)\big) \,;\, X\Big).$$

The shielded agent differs from the unshielded one by a single guarded choice. It is actuator aligned, because a guarded choice of two emitters is an emitter, so Lemma 4.2 applies to it unchanged and its epoch is the same three ticks.

Theorem 9.3 (Shields preserve controllable invariants)**.** *Let* $(Q, \mathsf{Act}, \mathrm{succ})$ *be a transition system over* $\mathcal{P}$*, let* Φ *be controllable with fallback* f*, let* E *be the sensor-aligned environment it induces, and let* π *be a deterministic policy. If the initial environment state* q_0 *lies in* Φ*, then the set of closed-loop states whose environment component lies in* Φ *is an invariant of* $\mathrm{Run}(A_\pi^\Phi, E)$*. Consequently every environment state reachable in the closed loop lies in* Φ*.*

Proof. Let Ψ be the set of states of $\mathrm{Run}(A_\pi^\Phi, E)$ whose environment component q lies in Φ. The initial state lies in Ψ because $q_0 \in \Phi$.

For closure, take a state in Ψ with environment component $q \in \Phi$ and consider one tick. By Lemma 4.2 clauses (1) and (2), the environment consumes τ on two ticks of every epoch and an action on the third. On a τ tick, Definition 4.1 keeps its state at q, so the successor is in Ψ. On the remaining tick it consumes the action emitted by the composite in that epoch, which by Definition 9.2 is $\pi(q)$ when $\mathrm{safe}(q, \pi(q))$ holds and $f(q)$ otherwise. In the first case $\pi(q) \in \mathrm{Safe}(q)$ by the test; in the second $f(q) \in \mathrm{Safe}(q)$ by the choice of fallback, which exists because Φ is controllable. Either way the consumed action a satisfies $\mathrm{succ}(q, a) \subseteq \Phi$, so every successor state has environment component in Φ and lies in Ψ.

Ψ therefore contains the image of the initializer and is closed under the transition, which is Definition 8.2 of Part I. Theorem 8.3 of Part I then places the reachable set of the closed loop inside Ψ, which is the second statement. □

Proposition 9.4 (Stochastic policies)**.** *Let* E *be a sensor-aligned environment over* $\mathcal{D}$ *whose kernel has support* succ*, let* Φ *be controllable for* $(Q, \mathsf{Act}, \mathrm{succ})$ *with fallback* f*, and let* $\pi : Q \to \mathcal{D}(\mathsf{Act})$ *be a rational stochastic policy. If* $q_0 \in \Phi$ *then every environment state reachable with positive probability in* $\mathrm{Run}(A_\pi^\Phi, E)$ *lies in* Φ*; in particular* Φ *holds along the whole run with probability one.*

Proof. Both the agent and the environment are over $\mathcal{D}$, so the closed loop is a $\mathcal{D}$-system and no statement here ranges over a combined model of $\mathcal{P}$ and $\mathcal{D}$; the nondeterministic system $(Q, \mathsf{Act}, \mathrm{succ})$ enters only through the hypothesis, as the support of the environment's kernel.

By Proposition 3.4, a state of $\mathrm{Run}(A_\pi^\Phi, E)$ has positive probability if and only if it is reachable in $\mathrm{Run}((A_\pi^\Phi)^\sharp, E^\sharp)$. Support abstraction commutes with the term structure: supp is a monad morphism by Lemma 3.2, so replacing every leaf step by its support turns A_π^Φ into the term $A_{\mathrm{supp}\pi}^\Phi$ over $\mathcal{P}$, in which the sensing leaf proposes some action in $\mathrm{supp}(\pi(q))$ and the same guarded choice follows; and it turns E into the sensor-aligned environment for succ. The proof of Theorem 9.3 used nothing about the proposing policy except that the emitted action passes the test or is the fallback, so it applies verbatim to $A_{\mathrm{supp}\pi}^\Phi$ and gives that every reachable environment state lies in Φ. Transporting back along Proposition 3.4 gives the statement, and a run leaving Φ would have to pass through a state of probability zero, so the set of such runs has probability zero. □

Corollary 9.5 (A shield that never fires is invisible)**.** *If* $\pi(q) \in \mathrm{Safe}(q)$ *for every* q *reachable in* $\mathrm{Run}(A_\pi, E)$*, then* $A_\pi^\Phi \sim A_\pi$ *on the reachable part, and the two closed loops are bisimilar.*

Proof. Relate each unfolding of A_π^Φ to the corresponding unfolding of A_π. The two agree on every tick except the emitting tick, where the guarded choice selects emit because the test holds at every reachable q by hypothesis, and emit is the emitter of A_π. The relation is a bisimulation, and Theorem 5.1 of Part II lifts it through the closed loop. □

The construction and the hypothesis are those of the shielding literature. A shield in the sense of Alshiekh and coauthors (2018) is a reactive system synthesized from a safety game that either supplies the learner with the set of safe actions before it decides, or corrects an unsafe action after it has decided; the winning region of that game is the greatest controllable predicate inside the specification, and Safe is the set of moves that stay inside it. Definition 9.2 is the second placement, and Theorem 9.3 is its correctness statement in the algebra: the safety requirement is Part I's invariant, the winning region is Part I's greatest invariant inside Φ, and the shield itself is Part II's guarded choice.

Example 9.6 (A corridor with a pit)**.** Let $Q = \{0, 1, 2, 3\}$ and $\mathsf{Act} = \{\mathrm{L}, \mathrm{R}\}$, with $\mathrm{succ}(q, \mathrm{R}) = \{\min(q+1, 3), \min(q+2, 3)\}$ and $\mathrm{succ}(q, \mathrm{L}) = \{\max(q-1, 0)\}$; the environment may overshoot to the right. Let $\Phi = \{0, 1, 2\}$, so that 3 is the pit. Then $\mathrm{Safe}(0) = \{\mathrm{L}, \mathrm{R}\}$ because $\mathrm{succ}(0, \mathrm{R}) = \{1, 2\}$, while $\mathrm{Safe}(1) = \mathrm{Safe}(2) = \{\mathrm{L}\}$ because $\mathrm{succ}(1, \mathrm{R}) = \{2, 3\}$ and $\mathrm{succ}(2, \mathrm{R}) = \{3\}$. Every state of Φ has a safe action, so Φ is controllable, and $f(q) = \mathrm{L}$ is a fallback. A policy that always proposes R is overridden at 1 and at 2 and passes at 0, and the shielded loop never enters the pit; the unshielded loop enters it on its second epoch. Controllability is not automatic: dropping L from the action set leaves $\mathrm{Safe}(1) = \mathrm{Safe}(2) = \emptyset$ and no shield exists.

The approval gate of Part III wraps an agent in the same way, with a human oracle in place of the safety test. The two constructions agree up to stuttering, and the exact sense in which they do explains both why the gate needs no controllability hypothesis and why it delivers less. Part III's human co-agent, its Definition 8.1, is an oracle when its answer is a function of the proposed symbol alone, and it sees only the action port, so a safety judgment that depends on the state can be asked of it only if the state is part of what is proposed. The construction below therefore uses the extended action alphabet $\mathsf{Act}^+ = Q \times \mathsf{Act}$, whose symbols name the state at which the action is to be taken; the environment applies the second component and ignores the first.

Proposition 9.7 (An approval gate is a shield with an idling fallback)**.** *Work over Act^+ and let E be a sensor-aligned environment, so that it is idle stationary. Call an output $d \in \Sigma$* inert *when the environment emits ε and stays on receiving it; the idle symbol τ is inert, and so is any reserved refusal marker the environment ignores. Fix an inert d. Let H be the human co-agent whose answer to the proposal (q, a) is ok when $a \in \mathrm{Safe}(q)$ and no otherwise, and which never answers pending; H is an oracle in Part III's sense, since its answer is a function of the proposed symbol. Write $\mathrm{Gate}_H^{\mathrm{link}}(A_\pi)$ for the agent over (I, Σ) obtained by linking H to the ask and answer ports of $\mathrm{Gate}_H(A_\pi)$, hiding them, and emitting d on the action port when the answer is no; the case $d = \tau$ is Part III's gate verbatim. Let idle be the emitter that emits d and halts, and let*

$$A_\pi^{\Phi,\tau} \coloneqq \mu X.\Big(\big(\mathsf{sense}_\pi \,;(\mathsf{emit} \oplus_{\mathrm{safe}} \mathsf{idle})\big)\,;X\Big), \qquad A_\pi^{\Phi,\tau,1} \coloneqq \mu X.\Big(\big(\mathsf{sense}_\pi \,;\mathrm{delay}_1\,;(\mathsf{emit} \oplus_{\mathrm{safe}} \mathsf{idle})\big)\,;X\Big).$$

Then $\mathrm{Gate}_H^{\mathrm{link}}(A_\pi)$ and $A_\pi^{\Phi,\tau}$ have the same traces after marker erasure and stuttering collapse, so each refines the other in Part I's sense. They are not bisimilar: the gate spends one tick per epoch putting its question, and the shield does not. If $q_0 \in \Phi$ then the environment states reachable in $\mathrm{Run}(A_\pi^{\Phi,\tau}, E)$ all lie in Φ, whether or not Φ is controllable.

Proof. The first step is a bisimulation with the delay-matched term. In $\mathrm{Gate}_H^{\mathrm{link}}(A_\pi)$ the running rule of the gate steps A_π once; on the tick at which A_π's emitter would emit (q, a) the gate emits τ on the action port and puts (q, a) to H, and on the following tick the waiting rule releases (q, a) on ok and emits d on no. In $A_\pi^{\Phi,\tau,1}$ the tick spent in delay_1 carries (q, a) unchanged and emits τ, and the next tick emits (q, a) or τ according to the test. Relate the gate state $(s, \mathsf{wait}(q, a))$ to the

state of $A_\pi^{\Phi,\tau,1}$ inside delay_1 carrying (q,a), the gate states (s,run) to the corresponding sensing and emitting states, and halted states to halted states. The waiting rule of Part III freezes A while the gate waits and delay_1 likewise holds its value for one tick, so related states carry the same data. The answer that resolves the wait is $\mathrm{safe}(q,a)$ by the definition of H, and safe is the test of the guarded choice, so the two agents take the same branch and emit the same symbol on every tick. The gate of Definition 8.2 of Part III does not undo the step that produced a refused proposal, and neither branch of the guarded choice revisits sense_π, so the policy state advances identically on both sides. The relation is a bisimulation containing the pair of initial states, so $\mathrm{Gate}_H^{\mathrm{link}}(A_\pi) \sim A_\pi^{\Phi,\tau,1}$.

The second step removes the delay. Each unfolding of $A_\pi^{\Phi,\tau,1}$ differs from the corresponding unfolding of $A_\pi^{\Phi,\tau}$ by one tick that consumes an idle input and emits τ, which is an idle position in Part I's sense. Deleting those positions carries every trace of $A_\pi^{\Phi,\tau,1}$ to a trace of $A_\pi^{\Phi,\tau}$ and inserting them carries every trace back, so the two have the same stuttering-closed trace sets and each refines the other. The consultation travels on a port that linking hides, so no marker output survives into the visible alphabet and marker erasure acts as the identity here. Composing the two steps, and using that bisimilarity implies equality of traces, gives the claim.

Bisimilarity fails for the pair as stated: the gate emits four symbols per epoch and the shield three, so on the input word that is idle throughout the two produce output words of different lengths, and no relation on states matches them tick by tick. Theorem 8.3 of Part III supplies the companion facts that a refused proposal never reaches the action port and that an always-approving oracle makes the gate transparent.

For the invariance, run the proof of Theorem 9.3 with idle in place of emit_f. In the branch where the test fails the composite emits d, which is inert, so the environment's state does not change and the successor stays in Ψ. No fallback action is needed, so controllability is never used. □

Remark 9.8 (What idling buys and what it costs). Proposition 9.7 identifies the price of the weaker hypothesis. A gate keeps the environment inside Φ without any assumption on Φ, because refusing to act is always safe against an idle-stationary environment. For the same reason it guarantees nothing about progress: a gate that denies every proposal produces a run in which the environment never moves, and that run satisfies the invariant. A shield with an action fallback needs Φ to be controllable, and in exchange the environment keeps moving, so a liveness argument about the fallback strategy is available. Against an environment that is not idle stationary, such as a plant that drifts while the operator deliberates, neither statement holds and the gate is not a shield.

10 Plans

Planning under nondeterminism distinguishes three kinds of solution, and the distinction is about which executions must reach the goal: some, all, or all fair ones. Read through the closed loop, the three are the three standard solution notions of a two-player reachability game, and the third is exactly the class of guarded recursions.

Definition 10.1 (Plan term)**.** Let $D = (Q, \mathsf{Act}, \mathrm{succ})$ be a nondeterministic planning domain, an AND/OR transition system in which $\mathrm{succ}(q,a)$ is a nonempty finite subset of Q. For $a \in \mathsf{Act}$ let $\mathrm{step}_a \coloneqq \mathsf{sense}_* \,;\, \mathsf{emit}_a$, where $\mathsf{sense}_* : 1 \Rightarrow Q$ reads the announcement and halts with the announced state and $\mathsf{emit}_a : Q \Rightarrow 1$ emits a. Let tests range over subsets of Q. *Plan terms* are generated by

$$p \;::=\; \mathsf{skip} \;\mid\; \mathrm{step}_a \;\mid\; p\,;p \;\mid\; p \oplus_b p \;\mid\; \mu X.\, p(X),$$

where every recursion is guarded. Each plan term denotes an actuator-aligned agent A_p by the operators of Part II. A memoryless plan is one of the form $\mu X.(\mathsf{skip} \oplus_g (\bigoplus_\sigma (\mathrm{step}_a)_{a\in\mathsf{Act}}\,;X))$ for a goal test g and a state-action table $\sigma : Q \to \mathsf{Act}$, with $\bigoplus_\sigma$ the policy choice of Definition 3.5.

Definition 10.2 (Outcome fairness)**.** An infinite epoch path $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} \cdots$ of $\mathrm{Run}(A_p, E_D)$ is *outcome fair* when, for every pair (q, a) occurring infinitely often on it, every element of $\mathrm{succ}(q, a)$ occurs as the successor infinitely often. A finite path ending in a state with no enabled action is outcome fair.

Outcome fairness constrains the environment's resolution of its own nondeterminism. It is a different condition from the scheduler fairness of Part I's observation policy, which constrains which component of a parallel composition is scheduled; neither implies the other, and only outcome fairness is used in this section.

Lemma 10.3 (Executions of a plan are paths of its closed loop)**.** *For every plan term p and every q_0, the set of epoch paths of $\mathrm{Run}(A_p, E_D)$ started at q_0 equals the set of executions of p in D from q_0, where an execution is defined by structural recursion on p:* skip *has the empty execution;* step_a *has the executions $q \xrightarrow{a} q'$ for $q' \in \mathrm{succ}(q, a)$; the executions of $p_1 \,;\, p_2$ are the concatenations of an execution of p_1 with one of p_2 from its final state; the executions of $p_1 \oplus_b p_2$ are those of p_i selected by b at the current state; and those of $\mu X.p(X)$ are the limits of the executions of its finite unfoldings.*

Proof. Induction on the structure of p, using Lemma 4.2 for the base case: A_{step_a} occupies one epoch, in which the environment announces q, the term emits a and the environment moves to some element of $\mathrm{succ}(q, a)$, so the epoch paths of the closed loop of step_a are exactly its executions. Sequential composition hands the halting value on with no tick spent and leaves the environment untouched, so epoch paths concatenate. Guarded choice consumes no tick and selects by the test on the value handed on, which by the base case is the announced state. For the recursion, guardedness makes every unfolding spend a tick, so the closed loop of $\mu X.p(X)$ is the limit of the closed loops of the finite unfoldings, and Theorem 7.1 of Part II identifies it as the unique such agent up to bisimilarity; bisimilar agents have the same epoch paths because bisimilarity implies trace equivalence, Theorem 4.4 of Part I. □

Proposition 10.4 (The three solution notions are the three game solutions)**.** *Let D be a nondeterministic planning domain, $G \subseteq Q$ a goal and p a plan term with initial state q_0. Consider the two-player reachability game on D in which the agent chooses the action and the environment chooses the successor, and read A_p as an agent strategy. Then:*

1. *p is a weak solution for G, meaning some execution of p reaches G, if and only if some path of $\mathrm{Run}(A_p, E_D)$ reaches G, that is, if and only if A_p witnesses the existential-path solution;*

2. *p is a strong solution, meaning every execution reaches G and terminates, if and only if every path of $\mathrm{Run}(A_p, E_D)$ reaches G, that is, if and only if A_p is a winning strategy in the reachability game;*

3. *p is a strong cyclic solution, meaning every fair execution reaches G, if and only if every outcome-fair path of $\mathrm{Run}(A_p, E_D)$ reaches G, that is, if and only if A_p is a winning strategy against a fair adversary.*

A memoryless state-action table is a strong cyclic solution if and only if the guarded recursion of Definition 10.1 built from it wins on every outcome-fair path.

Proof. Lemma 10.3 identifies the executions of p with the epoch paths of the closed loop, and reaching G is a property of that path, so the three equivalences are the three quantifier patterns applied on both sides of the same set of paths. The three notions on the left are those of Cimatti,

Pistore, Roveri and Traverso (2003), whose weak, strong and strong cyclic solutions require the goal on some execution, on every execution with termination, and on every fair execution respectively; the fairness they impose is that a state-action pair taken infinitely often exhibits each of its outcomes infinitely often, which is Definition 10.2. The game readings on the right are the standard ones: an existential path is a path in the AND/OR graph in which both players cooperate, a winning strategy for reachability is one against which no environment resolution avoids G, and a strategy winning against a fair adversary is one against which no outcome-fair resolution avoids G.

For the last sentence, a memoryless table is a plan term by Definition 10.1 and the recursion is guarded because step_a spends a tick; conversely the residual of that term at any point is determined by the current state, so it induces a memoryless table. The equivalence is then the case (3) already proved, applied to that term. □

Corollary 10.5 (Strong cyclic solutions win almost surely)**.** *Let σ be a memoryless table on a finite domain D and let E be any sensor-aligned environment over $\mathcal{D}$ whose kernel has support* succ*. Then σ is a strong cyclic solution for G from q_0 if and only if* $\mathrm{Run}(A_\sigma, E)$ *reaches G with probability one from q_0.*

Proof. Suppose σ is not a strong cyclic solution, so some outcome-fair path from q_0 avoids G forever. Let Y be the set of states it visits infinitely often; Y is nonempty because Q is finite, and $Y \cap G = \emptyset$. For $q \in Y$ the pair $(q, \sigma(q))$ occurs infinitely often, so outcome fairness puts every element of $\mathrm{succ}(q, \sigma(q))$ in Y. Hence Y is closed under σ, avoids G, and is reachable from q_0 along a finite prefix that has positive probability because each of its steps lies in the support of the kernel. Once inside Y the chain never leaves, so it never reaches G, and the probability of reaching G is at most 1 minus the probability of that prefix, which is less than one.

Conversely suppose σ is a strong cyclic solution. Then no σ-closed set avoiding G is reachable, by the argument just given read backwards: such a set would carry an outcome-fair path avoiding G, obtained by cycling through it so that every outcome of every infinitely repeated pair occurs infinitely often, which is possible because the set is finite and closed. So from every reachable state, G is reachable in at most $|Q|$ steps along a path of positive probability, and the minimum of these finitely many probabilities is some $\rho > 0$. The probability of failing to reach G in $n|Q|$ steps is therefore at most $(1 - \rho)^n$, which tends to zero. □

Example 10.6 (Weak but not strong cyclic)**.** Let $Q = \{\alpha, \beta, g, d\}$ and $\mathsf{Act} = \{\mathrm{try}, \mathrm{jump}\}$, with $\mathrm{succ}(\alpha, \mathrm{try}) = \{\alpha, \beta\}$, $\mathrm{succ}(\beta, \mathrm{try}) = \{g\}$, $\mathrm{succ}(\alpha, \mathrm{jump}) = \{\beta, d\}$, and g and d absorbing under every action. Let $G = \{g\}$. The table $\sigma_1(\alpha) = \sigma_1(\beta) = \mathrm{try}$ is a strong cyclic solution: the only path from α avoiding g repeats the pair (α, try) forever and takes only the α outcome, so it is not outcome fair. It is not a strong solution, since that path is an execution. The table $\sigma_2(\alpha) = \mathrm{jump}$, $\sigma_2(\beta) = \mathrm{try}$ is a weak solution, because the execution $\alpha \to \beta \to g$ reaches the goal, and it is not strong cyclic, because the execution $\alpha \to d \to d \to \cdots$ is outcome fair and never reaches g. Corollary 10.5 matches: under any positive-probability resolution, σ_1 reaches g with probability one and σ_2 reaches d with positive probability.

11 Linear feedback

The closed loop of a linear plant with a state-feedback controller needs no machinery beyond Part I. The plant is an environment, the controller is an agent over Id, and the epoch process is the difference equation of the textbook. What the algebra adds is an account of where the delay sits, and that turns out to matter.

Definition 11.1 (Linear loop)**.** Let $F \in \mathbb{R}^{n\times n}$, $G \in \mathbb{R}^{n\times m}$ and $K \in \mathbb{R}^{m\times n}$. The *plant environment* $E_{F,G}$ is the sensor-aligned environment over Id with $Q = \mathbb{R}^n$, $\mathsf{Act} = \mathbb{R}^m$, $\mathrm{succ}(x,u) = Fx + Gu$ and reward slot always $\bot$. The *controller* A_K is the actuator-aligned agent whose sensing leaf reads x and halts with $(x, -Kx)$ and whose emitter emits $-Kx$. The *linear loop* is $\mathrm{Run}(A_K, E_{F,G})$.

Proposition 11.2 (Stability of the linear loop)**.** *The epoch process of* $\mathrm{Run}(A_K, E_{F,G})$ *started at* x_0 *is* $x_{k+1} = (F - GK)x_k$*, and* $x_k \to 0$ *for every* $x_0 \in \mathbb{R}^n$ *if and only if* $\rho(F - GK) < 1$.

Proof. By Lemma 4.2 the action emitted in epoch k is computed from the state announced in that epoch, so $u_k = -Kx_k$, and the environment applies it to give $x_{k+1} = Fx_k + Gu_k = (F - GK)x_k$. Write $M = F - GK$, so $x_k = M^k x_0$.

If $\rho(M) < 1$, put $M = SJS^{-1}$ in Jordan form. A Jordan block J of size m with eigenvalue λ has $(J^k)_{ij} = \binom{k}{j-i}\lambda^{k-(j-i)}$ for $j \geq i$ and zero otherwise, and $\binom{k}{j-i} \leq k^m$, so every entry is bounded by $k^m|\lambda|^k$, which tends to zero because $|\lambda| < 1$. Hence $J^k \to 0$ for every block, so $M^k \to 0$ and $M^k x_0 \to 0$ for every x_0.

If $\rho(M) \geq 1$, choose an eigenvalue λ with $|\lambda| \geq 1$ and an eigenvector $v \in \mathbb{C}^n$, $v \neq 0$, and write $v = u + iw$ with $u, w \in \mathbb{R}^n$. Then $M^k v = \lambda^k v$, so $\|M^k u\| + \|M^k w\| \geq \|M^k v\| = |\lambda|^k \|v\| \geq \|v\| > 0$ for every k, and at least one of the real sequences $M^k u$ and $M^k w$ does not converge to zero. This is the discrete-time stability criterion of Astrom and Murray (2008), here obtained as a statement about the epoch process of a closed loop rather than as a separate theory. □

Example 11.3 (A double integrator)**.** Let $F = \left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$ and $G = \left(\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right)$. With $K = 0$ the closed-loop matrix is F, whose only eigenvalue is 1 with a single Jordan block, so $\rho = 1$ and the state grows without bound from a generic start. With $K = \left(\begin{smallmatrix} 1/4 & 1 \end{smallmatrix}\right)$ the closed-loop matrix is $\left(\begin{smallmatrix} 1 & 1 \\ -1/4 & 0 \end{smallmatrix}\right)$, whose characteristic polynomial is $(\lambda - 1/2)^2$, so $\rho = 1/2$ and every trajectory converges. With $K = \left(\begin{smallmatrix} 1 & 2 \end{smallmatrix}\right)$ the closed-loop matrix is $\left(\begin{smallmatrix} 1 & 1 \\ -1 & -1 \end{smallmatrix}\right)$, which squares to zero, so every trajectory reaches the origin after two epochs.

Remark 11.4 (One extra epoch of delay). The controller of Definition 11.1 computes its action from the state announced in the same epoch. A controller whose action lags by one epoch, $u_k = -Kx_{k-1}$, gives $x_{k+1} = Fx_k - GKx_{k-1}$, whose state (x_k, x_{k-1}) evolves by the block matrix $\left(\begin{smallmatrix} F & -GK \\ I & 0 \end{smallmatrix}\right)$. For the gains $K = \left(\begin{smallmatrix} 1/4 & 1 \end{smallmatrix}\right)$ of Example 11.3 that matrix has characteristic polynomial $\lambda^4 - 2\lambda^3 + 2\lambda^2 - \frac{3}{4}\lambda$ and spectral radius about 1.05, so the same gains that place the aligned loop at 0.5 make the lagged loop diverge. Epoch alignment is therefore not a bookkeeping convention: an implementation that reads a stale announcement is running a different and possibly unstable system, and Lemma 4.2 is what rules that out for the terms of this Part.

12 Limitations

The limitations of this Part are stated against the claim it contributes to, which is repeated here so that each restriction can be read directly against it.

> Part I fixes the object: discrete time, an effect monad T drawn from the list Id, $\mathcal{P}$, $\mathcal{D}$, and agents as effectful Mealy coalgebras with typed termination and failure as a terminal outcome. Parts II to IV introduce the operators under additional assumptions stated in each Part and prove, for each operator, that it is well-defined on agents, which relation it respects, and which laws hold at that relation: Part II for ;, $\oplus$, $\otimes$, $\triangleright$ and μ; Part III for Tr, channels, $\|$ and Gate_H; Part IV for Sup, Persist and Replay; Part V introduces

> no operator and interprets policies, plans, options, shields and controllers as terms of Parts I to IV, proving properties of their closed loops. For each of sixteen fields the series gives an explicit translation of one named mechanism into a term of the algebra together with an adequacy result for that translation, which is a soundness theorem, a one-way simulation, or a proved limitation; no field is claimed to reduce to the algebra.

This Part establishes the closed-loop half of that sentence for the terms of Sections 5 to 11, and only under five added assumptions. The interaction is epoch aligned in the sense of Definition 4.1. The decision models are finite, with rational parameters and a discount $\gamma \in [0, 1)$. Option durations have finite expectation. The shielded predicate is controllable and the environment is idle stationary. The plant is a discrete-time linear system with real matrices.

Epoch alignment is a hypothesis, not a theorem. Lemma 4.2 says what the epoch process of an aligned pair is; it says nothing about an agent and an environment whose phases do not match. Remark 11.4 shows that misalignment is not harmless, so a term that is not aligned needs its own analysis and inherits none of the results here.

The value results are about finite models with rational data. Theorem 6.4 and Theorem 8.6 both use boundedness of rewards to exchange expectation and summation, and Part I's monad $\mathcal{D}$ admits only finitely supported rational distributions, so continuous state spaces, continuous action spaces and irrational parameters are outside the object, not merely outside the proofs. Example 8.7 makes the point concretely for an observation model with irrational accuracy.

Theorem 7.3 assumes $\mathbb{E}[\tau] < \infty$. This is a real restriction: an option whose termination probability vanishes on a recurrent set of the induced chain never halts, the handoff of $A\,;B$ never occurs, and the right-hand side of the value equation has no meaning. The hypothesis is also not verifiable from the option's data alone, since it depends on the chain that the internal policy induces.

The shield results are about invariants, which are safety properties. Nothing here shows that a shielded agent still reaches a goal, and Remark 9.8 exhibits the extreme case in which the shield secures safety by never acting. A probabilistic shield, one that enforces a bound on the probability of violation rather than a hard invariant, is not covered: Proposition 9.4 still enforces the invariant on every positive-probability path, which is a strictly stronger requirement than a threshold and a strictly weaker construction than what a threshold would need.

Proposition 9.7 assumes a human oracle that is deterministic per state, never answers pending, and answers exactly the safety question. Each assumption fails in practice. An operator who answers differently on two occasions is not a function of the state; an operator who is away answers pending; and an operator who declines an action for a reason other than the stated invariant is not computing safe. The proposition then says nothing, and the gate is only what Part III proves it to be.

Planning is treated at the level of solution notions, not algorithms. Proposition 10.4 identifies three classes of plan with three classes of game strategy; it does not construct a plan, and the symbolic fixpoint computations that produce them are outside this Part. Corollary 10.5 is stated for finite domains and memoryless tables, and neither restriction is obviously removable: the closure argument uses finiteness of Q and the converse direction uses that the residual strategy depends only on the current state.

Continuous time is excluded by Part I. Proposition 11.2 concerns a sampled system whose sampling interval is one epoch, and the passage from a continuous plant to such a system, along with the error that passage introduces, is not treated anywhere in this series.

Learning is out of scope. Every policy here is fixed: a change of θ in π_θ, of K in a controller, or of π under policy improvement produces a different agent, and the map that produces it is an operation on agents rather than a term of the algebra. Corollary 6.5 is the closest this Part comes, and it treats improvement as an external map on the family $\{A_\pi\}$, not as a construction

inside the signature. The framework in which such maps are themselves morphisms, developed in the categorical cybernetics line by Capucci, Gavranovic, Hedges and Rischel (2021) and by Ghani, Hedges, Winschel and Zahn (2018), is the natural setting for that question, and nothing here contributes to it.

The claim about language models is narrow. Proposition 5.4 holds under the bounded-generation abstraction of Definition 5.1, and Remark 5.2 states what that abstraction rules out. A serving stack whose sampling distribution depends on the batch is not a kernel on contexts, and the proposition does not apply to it. The claim is also about one loop shape. ReAct is a single term; the many other loop shapes in use are terms too, but each needs its own translation and its own statement.

The code layer that accompanies this Part is a finite model. The rows of Appendix A are finite-model checks and exhaustive bounded checks over small state spaces and short tick budgets. They rule out the ways a definition can be wrong that a small example exposes. They are not verification, and every theorem above rests on its proof.

13 Conclusion

Policies, plans, options, beliefs, shields and controllers are terms over the operators that Parts I to IV of this series introduce, and the standard results about them are statements about the closed-loop semantics of those terms. A shield is one guarded choice; an option is a guarded recursion whose exit test reads a coin drawn while sensing; a plan is a term over sequencing, guarded choice and recursion; a belief agent is an agent whose state happens to be a distribution; a controller is a closed loop over the identity monad. The value equation of a Markov decision process, the semi-Markov equation of the options framework, the belief reduction, the correctness of a shield, the three solution notions of nondeterministic planning and the spectral stability criterion all follow from the epoch process that Lemma 4.2 identifies, together with hypotheses that this Part states rather than assumes.

The delay in Part I's closed loop is the finding that does not appear in any of the source formalisms. Feedback in this algebra passes through a register, so an agent acts on information that is one tick old, and a decision epoch is three ticks rather than one. Alignment repairs the mismatch and Lemma 4.2 shows that the aligned epoch process is the one-step process the textbooks describe, so nothing is lost. What is gained is that the misaligned case becomes visible and can be analyzed: Remark 11.4 exhibits gains that stabilize the aligned loop and destabilize the loop with one extra epoch of delay. The agreement of the approval gate with the shield whose fallback is idling has the same character. It explains why a gate needs no controllability assumption and why it cannot deliver progress, and neither half of that explanation is visible while the two constructions live in separate formalisms.

What remains open is the treatment of change. Every result here fixes the policy and studies the loop. An account of learning as an operation on agents, and of how such an operation interacts with the checkpointing and replay of Part IV, would require morphisms between agents of the kind the categorical cybernetics literature studies, and this series does not supply them.

References

[1] Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U. (2018). Safe reinforcement learning via shielding. *Proceedings of the AAAI Conference on Artificial Intelligence* 32(1). arXiv:1708.08611.

[2] Åström, K.J. (1965). Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications* 10:174-205. DOI: 10.1016/0022-247X(65)90154-X.

[3] Åström, K.J., Murray, R.M. (2008). *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press.

[4] Bellman, R. (1957). *Dynamic Programming*. Princeton University Press.

[5] Bertsekas, D.P. (2017). *Dynamic Programming and Optimal Control*, 4th ed. Athena Scientific.

[6] Capucci, M., Gavranović, B., Hedges, J., Rischel, E.F. (2021). Towards foundations of categorical cybernetics. *Applied Category Theory 2021*, EPTCS 372. arXiv:2105.06332.

[7] Cimatti, A., Pistore, M., Roveri, M., Traverso, P. (2003). Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence* 147(1-2):35-84.

[8] Ghallab, M., Nau, D., Traverso, P. (2004). *Automated Planning: Theory and Practice*. Morgan Kaufmann.

[9] Ghani, N., Hedges, J., Winschel, V., Zahn, P. (2018). Compositional game theory. *Proceedings of LICS 2018*, pp. 472-481. DOI: 10.1145/3209108.3209165. arXiv:1603.04641.

[10] Kaelbling, L.P., Littman, M.L., Cassandra, A.R. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101(1-2):99-134. DOI: 10.1016/S0004-3702(98)00023-X.

[11] Puterman, M.L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley.

[12] Sutton, R.S., Precup, D., Singh, S. (1999). Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112(1-2):181-211. DOI: 10.1016/S0004-3702(99)00052-1.

[13] Sutton, R.S., Barto, A.G. (2018). *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press.

[14] Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*, 2nd ed. Wiley.

[15] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y. (2023). ReAct: synergizing reasoning and acting in language models. *International Conference on Learning Representations 2023*. arXiv:2210.03629.

A Code evidence

The executable model that accompanies this series lives in `code/agent-algebra`, and the run recorded in `reviews/agent-algebra-vitest.txt` is the one the rows below cite. It explores finite state spaces to a stated cap under a stated tick budget, uses rational weights for $\mathcal{D}$ and finite nonempty subsets for $\mathcal{P}$, and never mixes the two instances in one check. The rows below are finite-model checks, meaning property runs over generated finite agents, and exhaustive bounded checks, meaning all inputs or all resolutions up to a stated bound. They are checks, not verification. Each theorem and proposition of this Part rests on the proof given with it; the checks rule out the errors a small example exposes. The separator > in a test name divides the file, the enclosing group and the case, and a label in the Claim column is written in its short form: `shield-adversarial` abbreviates `thm:decision-control:shield-adversarial`. Identifiers and file names are broken across lines where the column requires it. Test names are quoted exactly as the test suite declares them, so ordinary text such as `mu-term` and `F - GK` appears there in the form the suite uses rather than in this paper's mathematical notation.

Claim (label)	File and identifier	Test (file > describe > it)	Runs	What the test shows
Prop. 5.4, `llm-is-agent`	`src/envs.ts` `mockLLM`, `mockLLMStop`; `src/ops.ts` `react`, `fix`, `choiceB`	`test/part5-decision-control.test.ts` > `llm-is-agent` > `the seeded mock-LLM ReAct mu-term is bisimilar to a hand-written loop`	Finite-model check, 100 random kernels	A bounded-context kernel with rational finite-support sampling is an agent over $\mathcal{D}$ in Part I's sense, and the recursion of Definition 5.3 is bisimilar to the loop written by hand.
Thms. 6.4, 7.3, `closed-loop-value`, `options-compositional`	`src/mdp.ts` `mcValue`, `mcOption-Value`, `option-Bellman`, `value-Iteration`; `src/envs.ts` `smallMdp`	`test/part5-decision-control.test.ts` > `options-compositional` > `Monte-Carlo value of A ; B matches the option-level Bellman computation within tolerance`	Finite-model check, Monte Carlo with a stated tolerance	On a finite discounted model with γ in $[0, 1)$ the sampled value of the composite agrees with the value equation of Theorem 7.3 evaluated at the stopping time.
Thm. 8.6, `belief-agent`	`src/mdp.ts` `beliefUpdate`, `bruteBayes`, `beliefValue`, `mcBeliefValue`; `src/envs.ts` `smallPomdp`	`test/part5-decision-control.test.ts` > `belief-agent` > `belief update equals brute-force Bayes and DP value equals Monte-Carlo closed-loop value`	Exhaustive bounded check for the update; finite-model check for the values	The update of Definition 8.1 agrees with brute-force Bayes on a small model, and the value from dynamic programming agrees with the sampled closed-loop value.
Thm. 9.3, `shield-adversarial`	`src/ops.ts` `shield`, `choiceB`, `fix`; `src/envs.ts` `gridworld`, `proposeLeaf`, `applyLeaf`, `proposalSafe`	`test/part5-decision-control.test.ts` > `shield` > `shielded random policies produce 0 gridworld violations while unshielded produce more than 0`	Exhaustive bounded check over every enumerated path, $T = \mathcal{P}$ throughout	Every enumerated path of the shielded agent satisfies the invariant while the unshielded agent violates it, and the controllable-predecessor witness is checked at every state of the predicate.
Prop. 9.4, `shield-stochastic`	`src/ops.ts` `shield`, `choiceB`, `fix`; `src/envs.ts` `gridworld`, `proposalSafe`	`test/part5-decision-control.test.ts` > `shield` > `a stochastic shielded policy keeps the invariant on every sampled path`	Finite-model check, 200 seeds from non-colliding starts, $T = \mathcal{D}$ throughout	With a stochastic proposing policy every sampled path stays inside the predicate, which is the support statement the proposition makes. The comparison with the unshielded agent belongs to the preceding row, and the two effect instances are never mixed inside one term.

Claim (label)	File and identifier	Test (file > describe > it)	Runs	What the test shows
Prop. 10.4, planning-solution-notions	src/ops.ts fix, choiceB, seq; src/envs.ts gameLeaf; src/sim.ts allFair	test/part5-decision-control.test.ts > planning-solution-notions > a strong-cyclic mu-plan reaches the goal under all fair schedulers and a weak plan fails on some	Exhaustive bounded check over every schedule inside the tick budget	Every schedule is enumerated and the outcome-fair ones are characterised: the strong cyclic recursion reaches the goal on each of them, the two exceptions being the path on which the environment slips at every opportunity, which is unfair, and the path truncated by the budget; the weak plan fails on some path.
Prop. 11.2, linear-stability	src/linear.ts spectralRadius, simulateLinear, matSub, matMul	test/part5-decision-control.test.ts > linear-stability > the closed linear loop converges exactly when the spectral radius of F - GK is below 1	Finite-model check, more than 300 random real triples, borderline band excluded	The simulated loop converges exactly when the computed spectral radius is below one, with the band in which the radius is within 0.05 of one excluded because simulation cannot settle it; continuous time is out of scope and has no row.
Prop. 9.7, gate-is-shield	src/ops.ts gate, choiceB; src/envs.ts approverEnv, sayLeaf; src/equiv.ts eraseMarkers	test/part5-decision-control.test.ts > gate-is-shield > Gate with a safety-oracle human and the shielded agent have the same marker-erased, stutter-collapsed traces	Exhaustive bounded check	On one fixed three-action example, in which both sides report a refusal with the same inert symbol, the gate and the shield emit the same visible words and refine each other after marker erasure and stutter collapse. That is the relation Proposition 9.7 states, for the instance in which the inert refusal output is a reserved marker rather than the idle symbol. The bisimulation with the delay-matched term, the first step of the proof, is not checked here.