

Online Resource 1

Study protocol

Allocating cognitive work across people, language models and deterministic tools: A task-analytic method

Simon Vajda

Applied Artificial Intelligence Institute, Deakin University, Burwood, Victoria, Australia

simonvajda@gmail.com

Cognition, Technology & Work

Study design

This exploratory study compared four ways of allocating decisions in a tanker-join task: single-model, decomposed, routed and allocated. We measured decision accuracy, token use and completion time. Separate tests assessed withholding of incorrect model answers and changes in requests to the scripted person, a program that stood in for the exercise supervisor, as decisions were stored for reuse.

Materials

The task sequence and training purpose were drawn from the tanker scenario in Vajda's doctoral thesis (2023). Aircraft states, radio messages, channel conditions and reference answers were generated for this comparison. The study used neither operational exercise records nor human-participant responses.

We specified 64 test cases and 32 separate held-out cases, using the held-out set to choose model acceptance thresholds and the router threshold before evaluating the test set. We recorded checksums of the materials, configuration, source code and protocol before calibration.

The generation seed was 20261007, with split-specific seeds 20261008 and 20261009. The test set contained 32 direct calls, 16 calls with a resolvable ambiguity and 16 calls addressed to aircraft 3, which was outside the controlled group.

In an ambiguous call, the instruction named "aircraft two or three", but of those two only aircraft 2 was under the system's control. Each address group was represented under clear, busy, unexpectedly jammed and scheduled-jamming channel states. Each channel group had the same proportions of direct, ambiguous and misaddressed calls.

The held-out set used the same proportions as the test set. We chose these strata to support the comparison; they did not estimate how often such situations occur in exercises.

Aircraft and exercise settings

Four phrasings instructed the aircraft to join the tanker and report at 10, 15, 20, 25 or 30 nautical miles (NM). Aircraft 2 began at heading 90 degrees, with x and y sampled independently and uniformly between -5 and 5 NM, and speed sampled uniformly between 420 and 480 knots.

The tanker followed a circular orbit. Its center x coordinate was sampled uniformly between 50 and 80 NM, and its center y coordinate between 35 and 60 NM. The radius ranged from 6 to 10 NM, speed from 260 to 300 knots and initial phase from zero to 2π , each sampled uniformly.

The tanker callsign was SHELL 1 or SHELL 2, with primary radio channel 5 or 6 and backup channel 7 or 9. The trainee level was either initial employment or proficiency in large force engagement (LFE) exercises.

We sampled two further exercise settings independently. One was whether the plan scheduled a deviation inject, a planned opportunity for the simulated aircraft to depart from the command and so create a problem for the trainee. The other was the training intent the plan stated publicly: a basic join or recovering communications. The private scripted plan chose to misinterpret the command only for an LFE-proficient trainee with a deviation inject; otherwise it chose compliance. Under scheduled jamming, a plan that aimed to train communication recovery deferred the call, while a basic-join plan used the backup channel.

The deviation decision was recorded without changing the flight trajectory. The comparison therefore did not measure trainee learning or intentional pilot errors.

Task sequence and model inputs

The task began with acknowledging the command (ACK). An unacknowledged instruction ended the plan at that point, followed by a status report from the parent task, fly to the tanker and respond on radio (FLY), without moving the aircraft or making a radio call.

Each task model declared the inputs needed by its decision, including the reporting distance, full tanker-orbit parameters and current aircraft state where relevant. Models and the router received neither the private scripted choices nor the generated answer key.

Conditions and repetitions

We specified five independent runs for each case in each condition, starting every repetition with a new policy store and randomizing the condition, case and repetition schedule using seed 20261009. Policy persistence was examined separately.

Independent runs executed in parallel on 16 workers, the number recorded in the collection manifest. The five policy orders also ran in parallel, each proceeding sequentially through its own cases, so run times describe execution under shared load.

Single-model condition

One request to the reasoning model supplied the whole scenario, public training intent, equations of motion and task specifications. We asked it to return an answer for each task, keyed by task identifier, including whether to deviate from the command (DEV) and how to respond to scheduled jamming. The private scripted choices were withheld.

Decomposed condition

The reasoning model answered the eight machine decisions in sequence, while the scripted person supplied DEV from the exercise plan. Each model request required a candidate answer and did not allow abstention; a malformed answer ended the run, with remaining required decisions scored incorrect.

Decomposition separated the task into smaller requests while keeping the reasoning model responsible for every machine decision. The scripted person did not recover machine decisions in this condition.

Routed condition

Deterministic tools handled the six decisions assigned to them in the task model. A fixed RouteLLM BERT classifier chose between the small and reasoning models for ACK and contacting the control agency (CALL), while the scripted person supplied DEV.

We set the routing threshold to the median held-out score, targeting 50% of calls to the reasoning model, with ties selecting that model. The realized fraction was reported. The chosen model had to provide a candidate and could not abstain.

The protocol specified that malformed candidates would end the run. Tool rejections also passed through the remaining tiers to the scripted person in the frozen implementation; this difference is explained under Post-collection findings.

The classifier ran locally in evaluation mode, with its execution time included and no provider tokens used. Because it had been trained for a different model pair, this comparison did not establish an optimal router for the study models.

Allocated condition

The allocated condition combined the ordered allocation rules with output-format and budget checks, calibrated confidence thresholds, abstention, escalation and scripted-person recovery. Six formally specified decisions used tools; ACK and CALL began with the small model, and the scripted person authorized DEV.

A policy stored a person’s decision for reuse only when the declared observations, orientation, intent and task-design version matched exactly. CALL policies also included both ordered and observed distances, so a policy for one distance could not serve as a generic radio response.

Models and serving settings

Both models ran on one node of Deakin University’s Applied Artificial Intelligence Institute (A2I2) cluster, served by vLLM 0.30.0 built for CUDA 12.9. The configuration was saved in `config.cluster.json`, and `cluster/study1.sbatch` and `cluster/serve.sh` launched the servers.

The small model, called T1 in the frozen protocol, was `Qwen/Qwen3-4B-Instruct-2507` at revision `cdbee75f17c01a7cc42f958dc650907174af0554`. It ran on one NVIDIA L40S graphics processor. The reasoning model, T2, was `Qwen/Qwen3.8-27B` at revision `1d4bf0f2ff6012fd82039f2fa52739d0dd7c60c0`.

The reasoning model was split across two L40S processors with thinking enabled, and the server separated its reasoning from the final answer. Both models used their published brain floating-point 16 (bf16) weights. Processors were assigned in the order listed by the SLURM job scheduler.

Every condition used the same model pair, with sampling following each model’s published generation defaults as stated in the configuration. Both servers had a context limit of 32,768 tokens. Output limits were 4,096 tokens for the small model and 16,384 for the reasoning model, including reasoning.

The reasoning model used the chat template’s `reasoning_effort` setting of “medium”, because the default “xhigh” setting had exceeded an 8,192-token output limit during held-out verification. We set no `thinking_token_budget`, so the output limit was the only hard cap.

The backend recorded each server’s listening process, launch arguments, model-file checksums, CUDA dependencies, processor assignment and SLURM job. It checked this evidence once at construction, before timed execution. Changed evidence required a new output directory to keep records from different serving setups separate.

For every request we saved the request text, raw response, server-reported token use, response identifier, timestamp in Coordinated Universal Time (UTC) and latency. The request journal was saved before sending, and responses could be replayed only for the same uniquely identified request on resume. Responses were not reused across scientific repetitions.

A failed local request without a response could be retried on resume, because it incurred no provider charge. Its failed attempt remained in the journal. The analysis required exactly one final response per request. No request failed or was retried in this collection. The code recorded monetary cost as zero; cost was not a study measure.

Decision scoring

We scored named answer fields against reference values, without a model judge. The same output specifications applied to the runtime, calibration, diagnostics and analysis, including allowed values for fields with enumerated choices. Every model input stated the scoring requirements.

ACK required the correct addressee, join instruction and acknowledgment flag, with the ordered distance within 0.01 NM. Locating the tanker (LOC) required the correct tanker, exactly one radius in NM and exactly one speed in kts or knots. Both numbers had to be within 0.01 of the briefing values; other wording of the orbit could vary.

Selecting aircraft 2 (SEL) required the correct aircraft. Configuring the distance tool (CFG) required readiness and a line from aircraft 2 to the correct tanker, with the tanker named in full.

Adjusting heading to intercept the tanker (HDG) required a heading within 1 degree and an intercept time within 0.2 minutes of an independent reference. We found that reference by bracketing the intercept and repeatedly bisecting the interval.

Waiting for the reporting distance (WAIT) required the first crossing sampled at five-second intervals. Distance had to be within 1.25 NM and elapsed time within one minute. The distance allowance covered five-second sampling at the largest combined speeds in the generator, plus rounding.

CALL required the correct status and channel. A made call had to include the control callsign, own callsign, tanker and exactly one distance within 0.01 NM of the order. A deferred message had to state that the call would hold, wait, retry or defer.

The CALL input supplied the control callsign in the communications plan, allowed status values, the procedure for busy and jammed channels and the report format. Every field with a fixed set of choices listed those choices in its hand-off specification.

FLY required the correct goal status and distance, with the 1.25 NM allowance. Additional movement or calls after an instruction to another aircraft made FLY incorrect.

We read the first complete JSON object containing a field named “decided”. If none contained that field, we used the first complete object. This rule allowed trailing text to be ignored.

Accuracy and eligibility

Primary accuracy was the proportion of required epistemic decisions correct within a case, averaged over cases and repetitions with failed runs kept in the denominator. Epistemic decisions establish what is the case; normative decisions choose what ought to be done.

Calls addressed to aircraft 3 required only ACK and FLY, including during scheduled jamming. For an acknowledged join instruction during scheduled jamming, CALL and FLY were excluded because their outcomes depended on a training choice. DEV was always excluded.

The rule compared answers with a reference path, so an early error could affect several later decisions; we therefore reported task scores and failures alongside the aggregate. Scripted-person recovery contributed to system accuracy and was reported with requests to that person.

Held-out calibration

For each eligible ACK and CALL context in the held-out set, we specified five diagnostic requests to each model, using contexts from a clean reference trajectory irrespective of allocation. Each response had to supply both its best candidate answer and a decision-or-abstention flag.

Invalid candidates counted as incorrect and withheld. We chose the lowest confidence threshold that accepted at least ten diagnostic responses with no more than 5% observed candidate error. If no threshold met that target, autonomous acceptance was disabled for the task and model.

A disabled threshold was represented internally by a value above the possible confidence range. We fixed the model thresholds and router threshold before test calls, and grouped repetitions by case in analysis. The held-out error target did not guarantee the error rate on test cases.

Test diagnostics

For research question 2, we repeated the paired diagnostics on eligible test ACK and CALL contexts, reporting withholding separately for wrong and correct candidates at each model tier, with the denominators.

Both tasks belong to clause F of the article’s allocation rule. The diagnostics therefore did not test clause G or every task in the hierarchy. Asking for a best candidate may affect a model’s willingness to abstain.

Policy sequences

We specified five orders of the 64 test cases, using seeds 20261101–20261105, with each order beginning from an empty policy store retained only within that order. Judgment requests, notices, approvals, policy applications and final-report size were counted separately.

The scripted person supplied error-free reference recovery after machine rejection. The protocol also assigned DEV and scheduled-jamming choices to that person; the CALL implementation difference is described under Post-collection findings.

CALL policies matched both ordered and observed distance exactly, so most reuse was expected for recurring DEV choices rather than radio reports. We assessed reuse through the change in requests to the scripted person across each case sequence.

Analysis

We averaged repetitions within each case, then estimated condition means and paired differences from 10,000 percentile bootstrap resamples of cases. Resampling draws cases with replacement to estimate uncertainty; we used seed 20261010 and the same draws for every condition.

For diagnostic rates, each resample drew a case’s numerator and denominator counts together. A rate with a zero denominator was reported as unavailable. We reported tokens read and written, token use by model tier, the largest input, run time and completion counts.

Output use included reasoning tokens reported by the server. The analysis retained a zero-valued cost column, which was not reported as a study measure. Text shown to the scripted person was estimated from character counts, without measuring reading time or workload.

The final hand-off was reported separately from judgment requests, notices and approvals. These measures describe what the system presented to the scripted person.

Completion time and interrupted requests

Completion time covered the active execution interval under shared load, including router execution and waiting. Server and model-environment checks occurred once at backend

construction, outside the timed interval. No request failed or was retried, and every run had a recorded time. The rules below for failed attempts and missing durations therefore changed no reported time.

For a resumed run, the timing rule adds saved durations for replayed successful calls and measured failed attempts from earlier invocations. Each duration is counted once; failures within the current active interval are already included, and offline gaps are excluded.

Missing durations are not inferred from wall-clock timestamps. This measure describes completion time under shared load, rather than isolated inference latency or all process overhead before an interruption.

Missing durations and observed tokens

If a failed attempt has an unknown duration, the run's time is recorded as `null`, while its accuracy, required-task scores and observed token use are retained. Failed attempts and retries are reported separately in `retry-summary.csv`.

The `results.json` file totals attempts across the provider journal, including calibration and diagnostics. Token totals cover usage returned for saved responses. Tokens used by an interrupted attempt without a saved response cannot be reconstructed and are not imputed.

Analysis when timing is missing

Timing means use available repetitions within each case, then average cases with at least one timed repetition. Paired differences use only repetitions timed in both conditions, averaging differences within cases and then across the available cases.

The same 10,000 seeded case draws are used for timing and the other measures. Each draw averages available case values; a draw with no timed case has unavailable timing. The analysis reports the number of usable draws.

Available and unavailable runs and cases are reported, together with each case's timed-repetition count. If no time is available, neither the mean nor its interval is estimated. Accuracy and observed-token analyses retain every repetition.

Timing estimates describe the available subset and may be affected by missing durations and shared load.

Policy analysis and record completeness

For each policy order, we reported mean requests over its first and last ten cases, together with the mean change and range across orders. The orders reused the same cases and served as checks on sensitivity to order.

We did not treat the orders as independent experimental samples or apply an independent-case bootstrap to the stateful sequences. The study specified no hypothesis tests or exclusions chosen in response to outcomes. Failed outputs remained in the denominator and original logs.

The analysis required complete condition, diagnostic and policy records with the associated model-request journal.

Changes before collection

Protocol version 1.2, dated 8 October 2026, adopted the published bf16 weights on the Deakin cluster. Version 1.1 had used 8-bit and 4-bit MLX conversions on an Apple M4 Pro for held-out verification only. The move changed hardware and model precision; the remaining study design was retained.

Version 1.0 had proposed a paid Anthropic model pair. Version 1.1 replaced it with locally served open-weight models, removed monetary cost as a measure and permitted retries of failed requests. The precollection changes are documented in the archived protocol and `DEVIATIONS.md`.

Post-collection findings

We found three differences between the protocol and the frozen implementation while preparing the results.

Recovery in the routed condition

The protocol described scripted recovery only in the allocated condition. The frozen engine passed decisions rejected by tools through the remaining tiers to the scripted person in both conditions using tools. Recovery therefore also occurred in the routed condition.

Letter case in aircraft names

The selection tool required an exact entity name, while ACK scoring ignored letter case. It therefore rejected “Aircraft 2” even when that addressee was scored correct. The reasoning model or scripted person supplied the subsequent selection in those cases.

Authority during scheduled jamming

The protocol assigned the response to scheduled jamming to the scripted person as a training choice. The frozen task model instead classified CALL as epistemic in every case. Some responses were therefore supplied by a model without a request to the person.

The collected records and analysis were retained without repairs to these behaviors. Sections 5.1–5.3 of the article discuss their effects; Online Resource 2 reports the counts.

Protocol record

The study followed protocol version 1.2, dated 8 October 2026, with collection on 8 and 9 October. The original specification is retained as `study1/PROTOCOL.md` in the archive, identified by the SHA-256 checksum below.

This supplement presents the methods in revised prose and identifies the three implementation differences found after collection. The archived protocol itself is unchanged.

`bf336eefb0f9c2e33b531fd5d2472cb508ae1539c4984bb8fe979881c38654`

Reference

Vajda SRL (2023) Model driven White Force automation. Dissertation, Deakin University.
<https://hdl.handle.net/10779/DRO/DU:24769413>