NBER WORKING PAPER SERIES

CHAINING TASKS, REDEFINING WORK: A THEORY OF AI AUTOMATION

Mert Demirer
John J. Horton
Nicole Immorlica
Brendan Lucier
Peyman Shahidi

Working Paper 34859
http://www.nber.org/papers/w34859

NATIONAL BUREAU OF ECONOMIC RESEARCH
1050 Massachusetts Avenue
Cambridge, MA 02138
February 2026

We thank Seyed Mahdi Hosseini Maasoum, Guy Lichtinger, Anand Shah, Philip Trammell, Michael Zhao, and seminar participants at the Economics of AI Reading Group and Social Analytics Lab at MIT for helpful discussions. The views expressed herein are those of the authors and do not necessarily reflect the views of the National Bureau of Economic Research.

At least one co-author has disclosed additional relationships of potential relevance for this research. Further information is available online at http://www.nber.org/papers/w34859

Chaining Tasks, Redefining Work: A Theory of AI Automation
Mert Demirer, John J. Horton, Nicole Immorlica, Brendan Lucier, and Peyman Shahidi
NBER Working Paper No. 34859
February 2026
JEL No. D24, J23, J24, O33

ABSTRACT

Production is a sequence of steps that can be executed (1) manually, (2) augmented with AI, or (3) fully automated within contiguous AI-executed steps called "chains." Firms optimally bundle steps into tasks and then jobs, trading off specialization gains against coordination costs. We characterize the optimal assignment of humans and AI to steps and the firm's resulting job structure, showing that comparative advantage logic can fail with AI chaining. The model implies non-linear productivity gains from AI quality improvements and admits a CES representation at the macro level. Empirical evidence supports the model's key predictions that (1) AI-executed steps co-occur in chains, (2) dispersion of AI-exposed steps lowers AI execution at the job level, and (3) adjacency to AI-executed steps increases the likelihood that a step is AI-executed.

Mert Demirer
Massachusetts Institute of Technology
Department of Economics
and NBER
mdemirer@mit.edu

John J. Horton
Massachusetts Institute of Technology
MIT Sloan School of Management
and NBER
john.joseph.horton@gmail.com

Nicole Immorlica
Yale University
and Microsoft Research
nicimm@gmail.com

Brendan Lucier
Microsoft Research
brlucier@microsoft.com

Peyman Shahidi
Massachusetts Institute of Technology
Sloan School of Management
peymansh@mit.edu

1 Introduction

The proliferation of artificial intelligence (AI) tools is likely to raise productivity in the near term as existing tasks are automated. Although these gains may be substantial, the larger and likely longer run effects will come from reorganizing production, including shifts in the skill mix of tasks, human capital requirements, and the definition and design of jobs (Brynjolfsson et al., 2021). The workhorse framework in economics for studying the impacts of automation technologies is the task-based model, which represents production as the completion of a set of independent tasks (e.g., Autor et al., 2003; Acemoglu and Autor, 2011; Acemoglu and Restrepo, 2018, 2019, 2022). In these models, what matters is each task's amenability to substitution by, or complementarity with, the alternative to human labor (e.g., computers, robots, or AI). This approach, however, sets aside the fact that production requires tasks to be performed in some sequence, and that groups of tasks constitute "jobs." We argue that this sequencing is economically consequential and should be considered as it changes equilibrium predictions about which units of work are automated and how jobs are organized when technologies such as AI enter production.

We model production as a Leontief technology over an ordered sequence of exogenously specified *steps*, and treat the definition of tasks and their partitioning across jobs as endogenous outcomes. A *step* is the primitive unit of work in our framework, corresponding to what classic models would call a "task." We instead reserve the term *task* for a contiguous block of steps that the firm endogenously designates for joint execution by a worker. Jobs are then determined by how tasks are assigned across workers. In the absence of AI, tasks optimally collapse to single-step blocks so that steps and tasks coincide.

Firms seek to minimize production costs by choosing which steps of the sequence to delegate to AI, which to assign to human workers, and how to group tasks into jobs with different skill requirements. These choices trade off the benefits of specializing work among multiple workers against the coordination costs created by finer divisions of labor. In the absence of an effective AI technology, a firm might hire multiple high-skill workers in different job roles to accomplish distinct steps of its production process, incurring "hand-off" costs whenever work passes from one worker to another. As the AI technology improves, the firm might instead choose to automate many of these steps, lumping them together into one logical task, and then hire a single, potentially lower-skill worker to oversee this automation and validate the production outcome. Whether this strategy is optimal depends not only on the effectiveness of the AI technology at accomplishing individual production steps, but also on the sequential relationship between steps, the relative ease of verification versus manual execution, wages, and coordination costs between workers.

Compared to earlier waves of automation, AI is different in two related ways. First, its capabilities are broad and potentially relevant across many points in production, but also highly jagged, with sharp variation in performance even across adjacent steps in a workflow (Dell'Acqua et al., 2023). Second, getting value from AI often requires engineering the right context and interfaces, so the costs of sequencing work and handing off intermediate outputs become central. As a result, improvements in AI can shift deployment decisions at multiple points across the production chain at once, triggering discrete changes in overall production strategy. By endogenizing task sequencing and the bundling of steps into jobs, we capture these non-local effects of AI capability changes on task allocation and job design.

We distinguish the short run and the long run in our framework by the degree of flexibility in organizational adjustment, and study how AI affects production in each horizon. In the short run, human capital, wages, and job boundaries are fixed, and AI raises productivity by reducing the

time required to perform steps. In the long run, firms can reorganize their workflows by adjusting job boundaries, skill requirements, and AI deployment in tandem. Table 1 summarizes the key distinctions between the two horizons.

Table 1: Summary of AI's Impact in the Short and Long Run

Time Horizon	**Job Design**	**Worker Skills and Wages**	**AI Deployment Strategy**	**Comments**
Short run	Fixed	Fixed	Optimized within jobs	Productivity gains by automating existing tasks
Long run	Flexible	Skills and relative wages adjust to job design	Optimized across jobs	Joint optimization of AI deployment and job design to minimize total labor costs

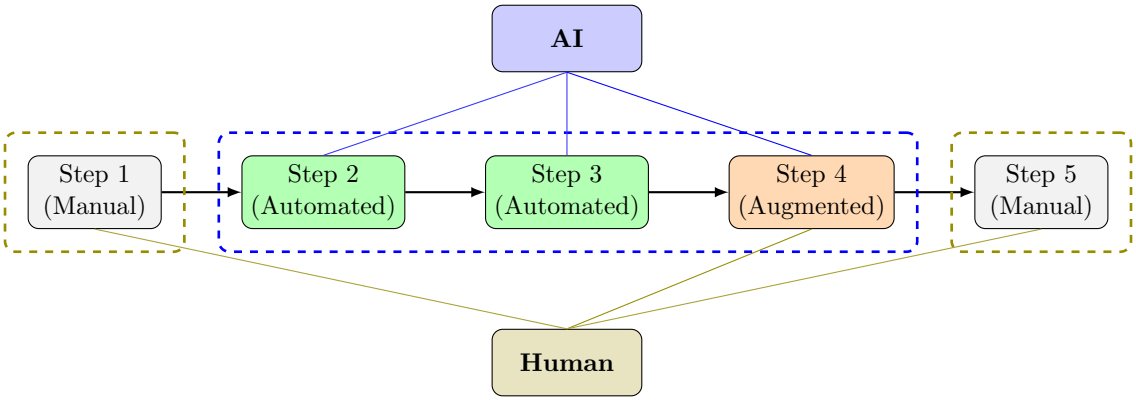
Automation versus Augmentation. Central to our framework is the distinction between full automation of a production step versus worker augmentation. In our model, three modes of step completion are recognized: manual, augmented, and automated. *Manual* completion of a step means that a human carries out the work without any AI assistance. *Augmented* completion involves a human performing the step with the use of AI, which might (or might not) reduce labor costs. For example, a human describes the requirements of a step to an AI tool, the AI then executes the step, and its output is reviewed and approved by the human.[1] In contrast, we say a step has been *automated* if it is completed by AI end-to-end without any direct human intervention. We view AI automation as a way of putting certain production steps "under the hood" of a logical meta-task that a worker is attempting to complete. To fix ideas, think of the workflow of a data scientist whose job requires completing five steps: defining the business question (Step 1), finding and fetching data needed for answering the question (Step 2), building an analysis pipeline (Step 3), drafting a report (Step 4), and presenting the findings (Step 5). As part of an AI-augmented request to complete one of these steps (such as Step 4: drafting a report), the AI might also be assigned to complete one or more preceding steps (such as Step 3: running an analysis, or Steps 2 and 3: finding data and running an analysis) as part of that single request. Any such preceding step is said to be automated, and the set of all such automated steps, plus the final step that the worker is actively engaged with, can be thought of as a single worker task which we term an *AI chain*. By choosing which steps to automate and which to assign either manually or via augmentation, the firm implicitly designs the set of tasks exposed to its workers.

Although augmented and automated production steps both involve AI, they differ in one crucial respect: AI augmentation demands direct human oversight of the AI's output, whereas automation does not. Figure 1 illustrates these concepts using the five-step data scientist workflow described earlier. Steps 1 and 5 are performed manually by a human. Steps 2, 3, and 4 form an AI chain and are done using AI. Steps 2 and 3 are automated because their outputs feed directly into subsequent AI steps without human review. Step 4 is augmented because its output is evaluated by a human before proceeding. The resulting worker tasks in this job thus become $1, 4$, and 5.

When a single step is performed by an AI, there is always a human requesting and evaluating its output in an augmented manner. But with two or more steps it is possible to re-configure production so that an AI completing one step passes its output directly into the next step without

[1] In our model, with highly capable AIs the human role shrinks to prompting and judging AI outputs, consistent with the view in Agrawal et al. (2019).

Figure 1: Illustrative Example for Division of Labor



Notes: This figure illustrates division of labor across five steps: Steps 1 and 5 are manual; Steps 2 and 3 are AI-automated (done without direct human intervention); Step 4 is AI-augmented (done with AI but requiring human oversight). Lines from the AI box indicate steps involving AI, and lines from the Human box indicate steps requiring human execution or oversight. Dashed boxes group steps according to their task assignments: Steps 1 and 5 form separate human tasks, while Steps 2–4 form an AI chain task.

human intervention (i.e., the first step is automated). This chaining is potentially a source of large productivity gains *but* it requires AIs that can perform each step with a high probability of success. In this sense, the model's basic setup shares conceptual similarities with the O-ring model of production (Kremer, 1993; Gans and Goldfarb, 2026).

The decision to deploy AI on a given production step compares the cost of manual execution to the unified AI-based execution cost. This is different from saying that steps get assigned to whichever factor of production has comparative advantage in that step. In fact, AI chaining can overturn standard comparative advantage logic in assignment because, when an AI chain is executed, a human worker verifies only the output of the last step of the chain (i.e., the augmented step). Output verification is therefore a fixed cost of the chain rather than a marginal cost that scales with each additional step. That is, appending a neighboring step to an existing AI chain adds no additional verification burden, while it may reduce the probability of AI's end-to-end successful completion of the extended chain. By contrast, assigning the neighboring step to a human terminates the chain and creates an additional human checkpoint, which entails another output verification (if the step is augmented with AI) or the cost of human execution (if the step is performed manually). When AI is sufficiently reliable on the marginal step, avoiding this additional human checkpoint can dominate, pulling the step into AI execution even if manual human execution is preferred for it in isolation. In the data scientist workflow example, the worker verifies the report once at Step 4, so verifying a report produced after AI executes Steps 3–4 requires the same effort as verifying a report produced after AI executes Steps 2–4; the only difference is that the AI is less likely to succeed at the latter on any given try. If AI's success probability at finding and fetching data is sufficiently high, Step 2 is appended to the AI chain to avoid creating an additional human checkpoint, even if the data scientist has comparative advantage in data collection in isolation.

More generally, we should expect the gains from AI automation to be greatest when automatable steps co-occur in the production process. For example, consider lecture-based teaching versus tutoring. In lecture-based teaching, many AI-suitable activities (e.g., background research, drafting

slides, generating examples) are clustered in a "preparation" block, making it feasible to delegate them to AI in a single chain and verify only the final output. In tutoring, by contrast, these activities are interleaved with diagnosis and rapid back-and-forth with a student, so automatable steps are more dispersed and delegating preparation activities to AI is less valuable.[2] Informally, we can think of a job as more or less "fragmented" in its AI exposure depending on whether the steps where AI is most effective are mutually coincident or dispersed throughout the workflow. In Section 5.1, we define a fragmentation metric for jobs and show analytically that it approximately tracks the impact of optimal AI deployment. In Section 7 we provide empirical evidence that jobs with higher fragmentation see a weaker translation from AI exposure to AI execution.

Jobs and Worker Skill. Each production step is associated with a time requirement and a skill requirement for manual execution, and a (potentially different) time and skill requirement for AI-augmented execution.[3] These cost parameters are exogenously determined and we treat them as given. To complete a job, a worker must possess the skills required of all tasks that make up the job description. Workers are ex-ante identical, as in Becker and Murphy (1992), and acquire the skills required for their assigned jobs after firms specify their roles. The total labor cost (i.e., wage bill) of a job increases with both the time needed to complete all steps and the skill level of the worker assigned to the job.

To capture the inefficiencies of fragmenting work into narrowly defined jobs, we introduce a *hand-off cost*, in time, that must be paid whenever output moves from one worker's job to another's. This creates a tension in job design: specialized jobs, containing fewer tasks, align worker skills more precisely with the job's requirements but incur higher hand-off costs. By contrast, generalist jobs, which bundle multiple tasks with different skill needs, reduce hand-off costs but require workers to acquire a broad set of skills, only a subset of which is being used at any given time. Appendix Figure D.1 illustrates this trade-off using a simple two-task production process.

AI adoption changes how tasks are defined and executed, which reshapes the time and skill requirements along the production process and therefore the forces that determine worker specialization. When certain tasks become automated, the optimal assignment of the remaining tasks across workers can shift, changing the role of skilled labor as an input to production and affecting wages as well as the distribution of productivity gains. We view this as a channel through which optimal AI deployment can shift the skill content of work, raising or lowering the relative demand for skilled labor within a given production process.

Optimizing Automation and its Implications. Assigning steps to production factors is not a completely trivial optimization problem, even computationally. For a firm with m steps in its production process, the number of possible production arrangements grows exponentially in m. For the short-run problem of optimizing AI deployment given fixed job responsibilities and worker skill levels, we show how to compute the optimal strategy for an m-step job in time $O(m^2)$ using dynamic programming. The firm's long-run optimization, which includes optimal assignment of steps and allocation of tasks to worker jobs, can also be solved in polynomial time, subject to an error term that can be made arbitrarily small. Although we do not do it in this paper, this

[2] Rather, we might expect AI exposure in tutoring to take the form of fully automated end-to-end AI tutors, once they become effective enough that it is feasible to chain all tutoring activities.

[3] AI automation incurs no human time or skill requirement.

algorithmic approach could be combined with micro-details of task content to create rich estimates of how technological change in various tasks would impact workers.

The AI chaining feature of the model implies that improvements in AI quality can generate non-linear effects on labor demand and wages. For any fixed production arrangement, better AI reduces costs smoothly by raising AI success probabilities and lowering expected execution times. However, firms optimize costs over a discrete set of AI deployment strategies and job designs, so the cost-minimizing arrangement can switch at particular AI quality thresholds. As a result, marginal improvements in AI may have little or no effect when they do not change the optimal production arrangement, which is especially likely in the early days of adoption when AI quality is low. Once AI quality crosses a reorganization threshold, longer AI chains and/or new job designs become viable and the optimal organization shifts abruptly, changing task allocation, job structure, and thus labor demand and wages. Our framework thus provides a microfoundation for the productivity J-curve phenomenon (Brynjolfsson et al., 2021) and yields testable predictions on when and how AI-driven productivity gains trigger structural reorganization in labor markets.

Micro Foundations of Aggregate CES Production. Our framework yields an algorithmic cost function at the firm level defined implicitly as the solution to a joint optimization over AI deployment and job design. This cost function in turn induces a Leontief production function in which output requires completion of many tasks, some executed manually and others with AI assistance. Drawing on classic results in production function aggregation (Houthakker, 1955; Levhari, 1968), we show that although each individual firm has a Leontief production, cross-firm heterogeneity in how firms deploy a commonly available AI technology allows aggregate production to admit a constant elasticity of substitution (CES) form at the macro level. We carry out this aggregation in a way that yields a macro production function with three inputs: economy-wide aggregate manual labor, aggregate AI-assisted labor, and aggregate capital. This representation links firms' internal organizational responses to AI to macroeconomic outcomes, while distinguishing manual and AI-assisted labor as distinct factors of production.

Empirical Evidence. We complement our theoretical analysis with new empirical evidence that speaks directly to the core mechanisms emphasized by the model. We assemble a dataset that links O*NET tasks to human assessments of AI exposure (Eloundou et al., 2024), realized AI execution outcomes from Anthropic's Economic Index (Handa et al., 2025), and GPT-generated workflow orderings for each occupation. This allows us to observe not only which production steps are exposed to AI, but also where they sit in an occupation's workflow and whether they are ultimately performed manually, augmented with AI, or automated by AI.

We test, and find empirical evidence consistent with, three key predictions that follow from the theoretical model. First, we examine whether AI-executed steps appear in contiguous blocks rather than being randomly scattered throughout the production workflow. Co-occurrence of AI steps is a direct implication of the model's chaining mechanism, in which the primary gains from AI arise when multiple adjacent steps are delegated jointly to AI. We document that in the data AI execution indeed operates over consecutive steps, a pattern consistent with modeling AI as acting on sequences rather than as an independent execution substitute at the individual step level.

Second, the model emphasizes that sequencing plays a central role in determining the returns to AI automation. We show empirically that, controlling for the share of steps exposed to AI, occupations whose AI-exposed steps are more dispersed across the production workflow exhibit a

substantially lower share of their steps executed by AI. This finding supports the fragmentation logic of the model and illustrates why considering just the share of exposed steps to AI can be misleading for predicting occupational impacts of AI when production steps are technologically interdependent.

Third, the chaining mechanism implies strong local complementarities in AI automation decisions within a workflow. When a step is positioned next to AI-executed neighbors, local gains from automating it as part of a chain may induce its AI execution even if human execution would be preferred for that step in isolation. We show empirically that when conceptually similar steps appear across different occupations, a given step is more likely to be executed by AI in the occupation where its neighboring steps in the workflow are also AI-executed. This pattern provides direct evidence that AI execution depends not only on comparative advantage and step-level characteristics, but also on the local production context created by adjacent steps in the workflow.

2 Related Literature

Task Interdependence in Task-based Models of Automation. Our framework relates to the literature on task-based models of technical change, which generally view tasks as predetermined and technologically independent objects combined through a CES production function (Acemoglu and Autor, 2011; Acemoglu and Restrepo, 2018, 2019; Acemoglu et al., 2024). In these settings, task-level comparative advantage pins down factor assignment and automation follows relative input efficiencies. We depart from this literature by modeling production as a sequence of technologically interdependent steps, where AI chaining creates complementarities across adjacent steps that can overturn comparative advantage as the driver of who does what.[4]

The chaining feature of the model further connects to, and helps reconcile, competing views about automation in production. Although much of the AI and labor literature emphasizes task-level substitution between AI and workers, Autor et al. (2003), Acemoglu and Restrepo (2019), and Bresnahan et al. (2002), among others, argue that meaningful substitution often occurs at the system level. Our model shows how step-level automation decisions can aggregate into system-level changes through AI chaining. Rather than requiring that steps be automated one by one, entire chains can be automated in a single leap, consistent with the perspective emphasized by Bresnahan et al. (2002).

Measuring AI Exposure and Realized Execution. A rapidly growing literature has sought to quantify the impact of new automation technologies on the labor market. Early work studies the susceptibility of occupations to computerization (Frey and Osborne, 2017), and more recent work develops measures of occupational exposure to AI (Brynjolfsson et al., 2018; Webb, 2020; Felten et al., 2021; Eloundou et al., 2024; Tomlinson et al., 2025), typically by mapping technologies to specific tasks and then aggregating these task-level mappings linearly to construct occupation-level indices. We depart from this literature by showing that linear aggregation obscures the crucial role of task

[4] A related perspective on task interdependence in task-based models appears in Trammell (2026), who studies learning spillovers across tasks. Although both his framework and ours feature interdependent tasks, their focus and margin of analysis differ sharply. In our setting, task interdependence is an intrinsic technological feature that creates coordination frictions when production is shared across multiple workers. By contrast, Trammell (2026) models interdependence as an endogenous result of economies of scale for the same worker, where performing more tasks raises productivity on subsequent tasks through learning, abstracting from cross-worker coordination frictions within the firm.

interdependence. In our framework, the productivity gains from AI depend not only on how many of an occupation's steps are exposed to AI, but also on where those steps sit relative to one another in the production sequence. To capture these non-linear interactions, we introduce the concept of "fragmentation," the degree to which AI-suitable steps are dispersed across the workflow. In this sense, our fragmentation index serves as a bridge between occupation-level exposure measures and the actual patterns of AI execution observed in practice.

Technology Adoption, Complementarities, and O-Ring Dynamics. Our work contributes to the literature on friction-laden technology adoption and organizational complementarities. Empirical work shows that information technologies raise productivity primarily when combined with complementary organizational changes (Bresnahan et al., 2002; Bloom et al., 2016). This aligns with the supermodularity framework of Milgrom and Roberts (1990, 1995), in which clusters of mutually reinforcing practices drive performance. We provide a micro-foundation for these complementarities through the lens of the O-ring theory of production (Kremer, 1993): when tasks are complementary because of sequencing, the returns to improving any one step depend on performance elsewhere in the chain. In our setting AI chaining makes this logic salient, since the payoff to automating a step depends on whether adjacent steps can be executed as part of the same AI block, generating threshold effects and discrete changes in optimal AI adoption and job design. Our results align with Gans and Goldfarb (2026), who likewise emphasize task interdependence in O-ring production and show that complementarities can generate "lumpy" automation decisions via time/attention reallocation rather than workflow adjacency.

Our framework also provides a micro-foundation for the "productivity J-curve" argument (Brynjolfsson et al., 2021), in which adoption costs and transitional reorganization precede realized productivity gains. In our model, the marginal benefit of improving AI technology grows non-linearly, with substantial gains emerging only once the technology crosses thresholds that induce discrete reorganizations of work and enable the formation of longer AI chains. This dynamic is consistent with recent analyses of U.S. Census microdata, which show that AI adoption initially reduces productivity (McElheran et al., 2025), as well as evidence from several other studies (Furman and Seamans, 2019; Tambe and Hitt, 2012; Bonney et al., 2024; McElheran et al., 2024).

Division of Labor and the Boundaries of the Job. Finally, our work connects to the literature on the division of labor within a firm. We apply the Coase–Williamson logic that firm boundaries are shaped by coordination, transaction, and governance costs (Coase, 1937; Williamson, 1971, 1979) to the "boundary of the job," where the firm chooses how many and which tasks to bundle into a worker's role, thereby determining the degree of worker specialization endogenously (Murphy, 1986). In our setting, hand-off costs serve as an intra-firm analogue of transaction costs: assigning a task to a different job moves it across a governance boundary and exposes it to coordination frictions, whereas integrating tasks into the same job resembles vertical integration at a finer, task-level margin. Our approach builds on the view that the division of labor is limited by coordination frictions and the difficulty of integrating specialized knowledge (Becker and Murphy, 1992), and it echoes Dessein and Santos (2006) in highlighting the tension between specialization and the need to coordinate interdependent tasks.[5]

[5]Ide and Talamàs (2025) provides a related view on how AI in particular affects the structure of work. While we examine how it redefines job boundaries through a task-based framework, they embed AI as an autonomous problem solver in a knowledge hierarchy model (Garicano, 2000; Garicano and Rossi-Hansberg, 2006) and study how

In our framework, AI does not alter the hand-off costs associated with steps, which are intended to capture coordination frictions arising from tacit or social knowledge that is not easily executable by AI (Deming, 2017). Nevertheless, AI can still restructure the division of labor through chaining in two distinct ways. First, on the intensive margin, chaining can pull previously manual steps "under the hood" of a single AI-executed task, narrowing the span of activities requiring direct human attention and shifting responsibility within a role. Second, on the extensive margin, by lowering the cost of linking adjacent steps that lie across a job boundary, chaining can make it optimal to redraw that boundary, reassigning activities across roles and altering the pattern of realized hand-offs and coordination within the firm. Both channels effectively alter the skill requirements of workers and the expertise they must possess to perform their jobs (Autor and Thompson, 2025).

3 Model

A firm's production process consists of a sequence of steps $\mathcal{S} = (s_1, \ldots, s_m)$.[6] The firm can partition contiguous subsequences of steps into a sequence of tasks $\mathcal{T} = (T_1, \ldots, T_n)$ where $T_b = (s_i, \ldots, s_{i+\ell})$ for some i and ℓ. The firm can also partition contiguous subsequences of the resulting tasks into jobs $\mathcal{J} = (J_1, \ldots, J_p)$. Figure 2 provides an illustration for a production process with $m = 7$ steps, $n = 5$ tasks, and $p = 3$ jobs. The cost of any such partitioning depends on the time and skill needed for the component steps, the mode in which the step is completed, and the grouping of steps and tasks. We discuss these in turn.

3.1 Steps

Each step can be completed either with AI assistance or manually (without AI). If the firm assigns step i to manual execution (denoted by M), it hires a human worker of skill level c_i^M and pays for the time t_i^M required to complete the step without AI. Alternatively, if the firm assigns step i to AI-assisted execution (denoted by A), it hires a human worker of (potentially different) skill level c_i^A and pays for the time t_i^A spent completing the step in collaboration with AI.[7] Such a collaboration succeeds independently with some (step-dependent) probability $q_i = \alpha^{d_i}$ where α is the quality of the general purpose AI technology and d_i represents how "difficult" step i is for AI. The collaboration must be repeated until it succeeds, incurring the time cost of t_i^A per iteration for a total expected time cost of t_i^A/q_i.

Definitions 1 and 2 formalize the two modes of performing a single step explicitly.

Definition 1 (Manual Step)**.** *A step i is said to be performed manually if it is executed entirely by a human worker without AI assistance. The associated skill and time costs for a manual step are denoted by (c_i^M, t_i^M).*

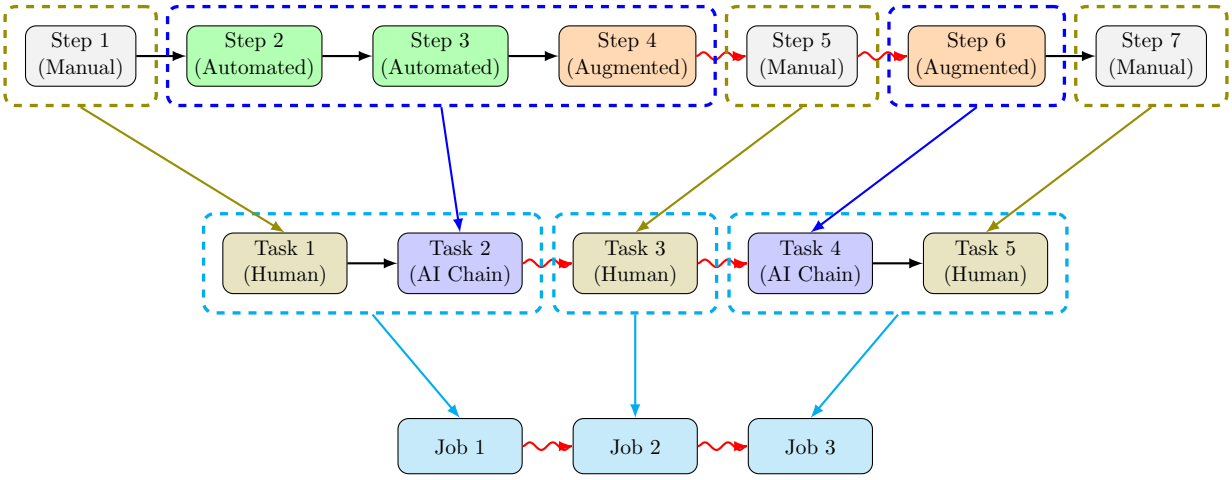
Definition 2 (Augmented Step)**.** *A step i is said to be augmented if it is executed by the AI, after which its output is reviewed and approved by a human worker. The associated skill and (expected) time costs of managing a single augmented step are denoted by $(c_i^A, \frac{t_i^A}{q_i})$, where $q_i \in (0, 1]$ is the AI's probability of successfully completing the step.*

it reshapes spans of control within the firm.

[6] We hold the set of production steps fixed and abstract from the creation or disappearance of steps in our model.

[7] This could, for example, represent the time spent formulating a prompt for AI and checking the response, or "managing" the AI.

Figure 2: Illustrative Example of a Firm's Production



Notes: This figure illustrates how a firm aggregates steps into tasks and tasks into jobs. The top layer shows $m = 7$ steps executed in manual (gray), automated (green), or augmented (orange) modes, with dashed boxes indicating task groupings. The middle layer aggregates steps into $n = 5$ tasks, classified as human-executed tasks (olive) or AI chains (purple), with cyan dashed boxes indicating job groupings. The bottom layer aggregates tasks into $p = 3$ jobs (cyan), each assigned to a separate worker. Vertical colored arrows show aggregation across layers, and horizontal arrows show production flow within each layer. Red curly arrows mark hand-off costs incurred at job boundaries, and can be traced from the bottom layer to the corresponding tasks and steps above.

3.2 Tasks

Tasks are the basic unit of work of a human worker. Any step performed in isolation during the production process, either manually or in collaboration with AI augmentation, is a task. The associated skill and time costs for the resulting task are precisely the skill and time costs for the associated step performed in the chosen mode. Steps can also be automated by chaining them together, creating a new aggregate task.

Definition 3 (Automated Step)**.** *A step is said to be automated if it is executed entirely by AI without direct human intervention, and its output is passed directly to a subsequent augmented or automated step. The direct human costs (in both skill and time dimension) associated with an automated step are zero.*

Definition 4 (AI Chain)**.** *An AI chain is a contiguous block of one or more sequential steps executed by AI, in which all steps but the final one are automated and the final step is augmented. An AI chain spanning steps $(s_\ell, \ldots, s_r)$ has steps $(s_\ell, \ldots, s_{r-1})$ automated and step s_r augmented. The skill and (expected) time costs of this AI chain are given by:*

$$\left(c_r^A, \frac{t_r^A}{\prod_{i=\ell}^{r} q_i} \right),$$

where q_i denotes the AI success probability for step s_i.

We can think of an AI chain as a single aggregate task that is being delegated to an AI. Successful completion requires that the AI complete all steps in $(s_\ell, \ldots, s_r)$. The human worker who manages the AI execution of this chain need only prompt for and evaluate the goal step, s_r; all other steps in the chain are assumed to be fully automated "under the hood" and hence beneath the awareness of the human worker. Managing an AI attempt at the chain therefore has the same costs as augmenting step s_r: namely, skill cost c_r^A and time cost (per attempt) t_r^A. Successful completion of the chain requires that the AI complete *every* constituent step successfully. Assuming independent failures, this occurs with probability $\prod_{i=\ell}^{r} q_i$, which we can view as the success probability for the aggregate chained task. Putting this all together yields skill and expected time costs $(c_r^A, t_r^A / \prod_{i=\ell}^{r} q_i)$ for the AI chain, as described in Definition 4.

Note also that since $q_i = \alpha^{d_i}$, where recall d_i is a measure of the difficulty of step s_i, the probability of successful completion of an AI chain spanning steps $(s_\ell, \ldots, s_r)$ can be written as $\alpha^{(\sum_{i=\ell}^{r} d_i)}$. We can therefore think of $\sum_{i=\ell}^{r} d_i$ as the total difficulty of the chain, which aggregates additively over its constituent steps.

To summarize, a task is either a manually-performed step, an AI-augmented step, or an AI chain. We define an *AI deployment strategy* (or *AI strategy* for short) as a sequence of tasks $\mathcal{T}$ that partitions the step sequence $\mathcal{S}$ into contiguous subsequences, with each singleton task being designated for either manual or AI-augmented execution and non-singleton tasks being AI chains.

3.3 Jobs

A job is a subsequence of tasks that are assigned to a single human worker. Recall that each task T_b in task sequence $\mathcal{T} = \{T_1, \ldots, T_n\}$ has associated skill and time costs (c_b, t_b) that can depend on whether the task is being completed manually or with the aid of AI. A job J is a contiguous subsequence of tasks, say $J = (T_b, \ldots, T_{b+\ell})$ for some b and ℓ. We say that a partition $\mathcal{J} = (J_1, \ldots, J_p)$ of all tasks into jobs is a *job design*.

The firm hires one worker for each job in its job design $\mathcal{J}$. The worker for job J_j completes all tasks associated with their job. The total time needed to complete all tasks in job J_j is $\sum_{T_b \in J_j} t_b$.

A worker's wage is determined by the total skill required to complete their job. The total skill required to complete the tasks in job J_j is $\sum_{T_b \in J_j} c_b$. We then assume that a worker's wage is proportional to the skill level of their job.[8] That is,

$$\text{Wage}_j = \sum_{T_b \in J_j} c_b. \tag{1}$$

The firm must pay workers their wage per unit of time regardless of which tasks they are assigned to complete at any given moment.[9] While we allow the firm to specialize its job design and partition its production process into distinct jobs, we acknowledge that there are inherent inefficiencies and

[8] One motivation for this formulation is workers paying a human capital investment to acquire the skills necessary for a job. This investment must be offset by wages, and we express skill levels in units of their corresponding requisite wage. Rather than formalizing such a wage model here, we impose this as an assumption and provide a more detailed wage formulation in Section 6.

[9] This derivation implicitly assumes a common base wage rate; i.e., all wage differentiation is driven by the human capital cost of acquiring skills. More generally, different tasks T_b within job J_j could have different base wage rates w_b that could be influenced, for example, by the supply of and demand for labor of the corresponding type. One special case, explicitly motivated and discussed in Section 6, assumes manual tasks are executed by human labor at base wage rate w_M, and AI-assisted tasks are executed by AI management labor at base wage rate w_A. The formulation in Equation 1 implicitly normalizes these base wage rates to 1, incorporating them into skill costs c_b for

frictions that are introduced when splitting work among multiple workers. Absent such frictions in our model, it would always be optimal for a firm to specialize its workers as much as possible, assigning a distinct worker to each task of its production process. We therefore introduce *hand-off costs* to a job design, as follows. In addition to the time required completing tasks, a worker may need to spend time handing off their work to another worker if J_j is not the final job in the job design $\mathcal{J}$. We will assume throughout that the hand-off time of a worker assigned to job J_j depends only on the final step of job J_j (see red curly arrows in Figure 2). That is, conditional on where the hand-off occurs in the production process, its cost does not otherwise depend on previous hand-off events. Given step s_i, we write t_i^H to denote the additional hand-off time spent by a worker for whom the last step of their job is s_i. For notational convenience we define the hand-off time for the final step s_m to be zero: $t_m^H = 0$. Also for notational convenience, we will write $t^H(J_j)$ to denote the hand-off time of job J_j, which recall is equal to t_i^H if s_i is the final step in job J_j. Thus the total time needed to complete job J_j, including hand-off costs, is

$$t^H(J_j) + \sum_{T_b \in J_j} t_b.$$

Taking into account both time and skill requirements, the total wage bill paid to a worker assigned to job J_j, per unit of output, is

$$\text{WageBill}_j = \left(\sum_{T_b \in J_j} c_b \right) \left(t^H(J_j) + \sum_{T_b \in J_j} t_b \right). \tag{2}$$

Equation (2) highlights two opposing forces that shape a job's boundaries. Adding more tasks to a worker's job increases the cumulative skill requirement and thus the wage rate for that worker. However, if tasks are kept separate, workers must incur additional hand-off costs at each job boundary. These two effects—higher wages from combining tasks versus higher hand-off costs from keeping them separate—jointly determine the optimal degree of task aggregation. We explore this trade-off in greater detail in Section 3.5.

3.4 Firm's Organizational Structure

Given the sequence of steps $\mathcal{S} = \{s_1, \ldots, s_m\}$, the firm can design both the partition $\mathcal{T}$ of steps into tasks and the partition $\mathcal{J}$ of tasks into jobs. The choice of $\mathcal{T}$ determines the time and skill requirements of each task. Given $\mathcal{T}$, the choice of $\mathcal{J}$ then determines worker wage rates and time needed per unit of output (including hand-off costs). Formally, we can write $\mathcal{P}(X)$ for the set of partitions of a sequence X into contiguous subsequences. Then the full optimization problem faced by the firm can be expressed as

$$\min_{\mathcal{T} \in \mathcal{P}(\mathcal{S})} \min_{\mathcal{J} \in \mathcal{P}(\mathcal{T})} \text{TotalCost}(\mathcal{J}; \mathcal{T}) = \sum_{J_j \in \mathcal{J}} \text{WageBill}_j = \sum_{J_j \in \mathcal{J}} \left[\left(\sum_{T_b \in J_j} c_b \right) \left(t^H(J_j) + \sum_{T_b \in J_j} t_b \right) \right] \tag{3}$$

where recall that, for each $T_b \in \mathcal{T}$, t_b and c_b are as described in Section 3.2 and can depend on the selected mode of operation for individual steps.

notational simplicity. We maintain this normalization throughout subsequent sections until we explicitly distinguish human labor and AI management labor base wage rates in Section 6.

We refer to (3) as the firm's *long-term* optimization problem, as it anticipates the adjustment of worker wages to fit the skill requirements of each job and sets job responsibilities accordingly. We also define a *short-term* optimization exercise in which jobs and worker wages are fixed but the firm may still wish to use AI to maximize worker productivity. This is equivalent to minimizing the time needed for a worker to perform a unit of work in their assigned job. It therefore suffices to optimize for each job separately. Thus, in the *short-term* optimization problem, we can assume without loss of generality that the sequence of steps $\mathcal{S} = (s_1, \ldots, s_m)$ is to be completed by a single worker paid at a (normalized) unit wage. The resulting optimization problem faced by the firm can be expressed as

$$\min_{\mathcal{T}} \sum_{T_b \in \mathcal{T}} t_b \tag{4}$$

where recall that if $T_b = (s_\ell)$ is a manual task then $t_b = t_\ell^M$, and otherwise if $T_b = (s_\ell, \ldots, s_r)$ for some $\ell \leq r$ then $t_b = \frac{t_r^A}{\prod_{i=\ell}^r q_i}$ where q_i is the AI's probability of successfully completing step s_i.

3.5 Hand-off Costs and the Limits of Worker Specialization

Here we discuss how hand-off costs determine the optimal organization of production and limit full worker specialization in the firm's long-run problem (3). We leverage insights from a numerical example as well as a geometric illustration, which together show how task aggregation balances higher wages against lower coordination costs.

Consider a production process that, under a fixed AI deployment strategy $\mathcal{T}$, has $n = 3$ tasks with the following cost parameters:

Task	c_b	t_b	t_b^H
1	3	1	3
2	1	2	0.5
3	2	2	0

With three tasks, there are four ways the firm can design jobs.[10] Denoting jobs by square brackets, with three tasks the four possible job designs are $\{[1][2][3], \quad [1,2][3], \quad [1][2,3], \quad [1,2,3]\}$. The cost of these job designs for the example above is given in Table 2. In the absence of hand-off costs in Panel (a), the optimal job design corresponds to full specialization: each task forms a separate job assigned to a different worker. When hand-off costs are introduced in Panel (b), however, the optimal design changes to one in which tasks 1 and 2 are combined into a single job, while task 3 remains separate. This structure is optimal because it avoids the large hand-off cost that would otherwise arise between tasks 1 and 2 due to the large hand-off time value $t_1^H = 3$. By aggregating tasks 1 and 2, the firm pays a higher wage to a more skilled and more versatile worker but eliminates the costly coordination between the two tasks, lowering total production cost. Equation (2) captures this trade-off explicitly: adding more tasks to a worker's job raises the wage rate through the skill term $\sum_{T_b \in J_j} c_b$, while keeping tasks separate raises total cost through additional hand-offs $t^H(J_j)$. These two effects jointly determine the optimal degree of task aggregation.

[10] More generally, a production process with n tasks has $2^{(n-1)}$ unique job designs.

Table 2: Role of Hand-off Costs in Organizational Structure

Panel (a): Without Hand-off Costs

Job Design	**Job**	**Job Tasks**	$\sum b$	$\sum t_b$	**Job Cost**	**Total Cost**	**Optimal Design**
[1][2][3]	1	$\{1\}$	3	1	3	9	✓
	2	$\{2\}$	1	2	2		
	3	$\{3\}$	2	2	4		
[1,2][3]	1	$\{1,2\}$	4	3	12	16	
	2	$\{3\}$	2	2	4		
[1][2,3]	1	$\{1\}$	3	1	3	15	
	2	$\{2,3\}$	3	4	12		
[1,2,3]	1	$\{1,2,3\}$	6	5	30	30	

Panel (b): Including Hand-off Costs

Job Design	**Job**	**Job Tasks**	$\sum b$	$t^H + \sum t_b$	**Job Cost**	**Total Cost**	**Optimal Design**
[1][2][3]	1	$\{1\}$	3	4	12	18.5	
	2	$\{2\}$	1	2.5	2.5		
	3	$\{3\}$	2	2	4		
[1,2][3]	1	$\{1,2\}$	4	3.5	14	18	✓
	2	$\{3\}$	2	2	4		
[1][2,3]	1	$\{1\}$	3	4	12	24	
	2	$\{2,3\}$	3	4	12		
[1,2,3]	1	$\{1,2,3\}$	6	5	30	30	

Notes: This table reports total production costs for each possible job design in an example production process with a fixed automation strategy and $n = 3$ tasks. The cost parameters of the tasks are given by $(c_b, t_b, t_b^H) = \{(3,1,3),(1,2,0.5),(2,2,0)\}$. In the absence of hand-off frictions, full specialization, corresponding to job design [1][2][3], is cost-minimizing. Once hand-off costs are taken into account, the optimal organizational structure changes to job design [1,2][3], which avoids the large coordination cost between tasks 1 and 2 at the expense of employing a higher-cost (more skilled) worker to complete those tasks.

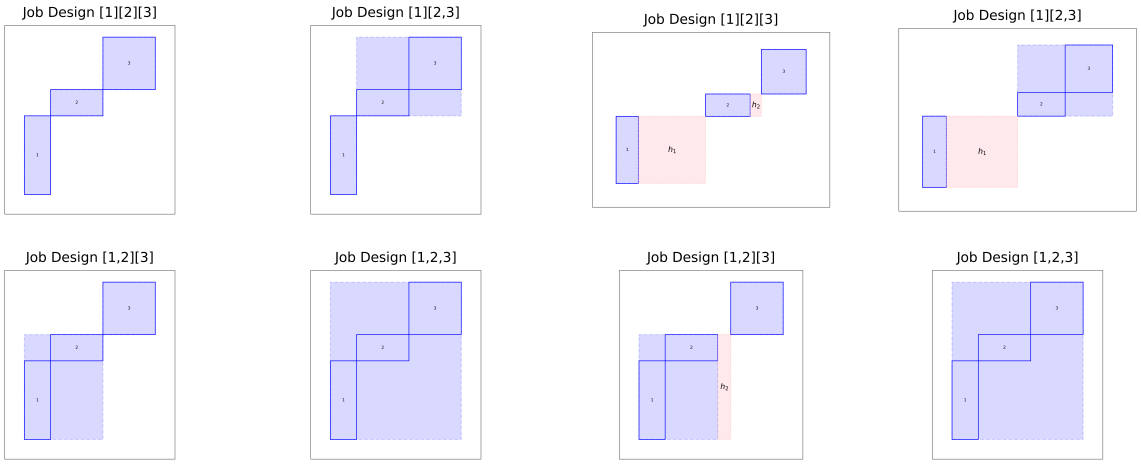
The cost of different job designs admits a geometric interpretation as well. Figure 3 visualizes the production process of the example in Table 2 as stacked rectangles, where each rectangle's height corresponds to c_b and its width to t_b. Panel (a) depicts the job design when hand-off costs are zero, Panel (b) shows the same process with hand-off costs. Visually, optimal job design balances two opposing forces. Combining tasks into a single job eliminates the pink rectangles representing additional hand-off costs between tasks, but creates inefficiencies due to the higher skill level of the worker required to do the job. Conversely, splitting tasks into separate jobs eliminates the intra-job inefficiencies but incurs hand-off time costs at each job boundary. These boundary costs scale with the cumulative human capital of the switching worker, so the height of the pink hand-off rectangle in Panel (b) equals the sum of human capital heights for that job. Cost-effective job design must balance these effects.

We have modeled hand-off costs as fixed and not influenced by the introduction of AI. In

Figure 3: Geometric Interpretation of Job Designs

(a) Job Design Without Hand-off Costs

(b) Job Design With Hand-off Costs



Notes: Each panel visualizes the production process as stacked rectangles, where the area of each job's bounding box represents its wage bill. Panel (a) sets hand-off costs to zero, in which case it is optimal to assign one task per job. Panel (b) introduces hand-off costs, which alters the cost structure and makes the [1,2][3] design cost-minimizing.

reality, AI may reduce the cost of hand-offs between workers as well. This might occur because the communication burden of handing off tasks could be partially automated through AI assistance. While we do not model this explicitly, we expect that such a force would reduce the impact of hand-off costs and result in a higher degree of worker specialization.[11]

4 Optimization

In this section we consider the optimization problems (both short-term and long-term) faced by a firm choosing how to integrate the AI technology into their production process. We begin by studying the short-term problem of optimizing AI deployment strategy keeping job design and worker wages fixed. We then move on to the joint optimization of deployment and job design taking into account long-term wage impacts.

4.1 Short-Term Optimization: AI Deployment Design

We begin with a "short-term" optimization problem: the job design and worker wages are assumed to be fixed, so the goal is to find the choice of AI strategy for a single job that minimizes the total time cost. This optimization problem analyzes short-run benefits of AI: where worker skills are

[11] A useful interpretation of hand-off costs is that they consist of two components: (i) peripheral coordination work that can be facilitated by AI (e.g., drafting messages, summarizing prior work, formatting documentation), and (ii) a purely manual remainder that relies on tacit, interpersonal, or social knowledge and is not easily executable by AI (Deming, 2017). For tractability, we abstract from the first component in our framework and treat its benefits as absorbed into execution and AI management time costs. The hand-off cost in the model should therefore be read as capturing the second, irreducibly human component, even though AI-facilitated reductions in the peripheral component may matter in practice.

fixed but AI can still be employed to speed up tasks through some combination of augmentation and automation.

Since the job design and worker wage are fixed, it suffices to optimize for the amount of time spent to complete a unit of work for each job separately. So, from this point onward, we assume that there is a single job J with m steps $\mathcal{S} = (s_1, \ldots, s_m)$, and our goal is to find the sequence of tasks $\mathcal{T}$ that minimizes total completion time. It turns out that the time-optimal production strategy can be calculated via dynamic programming in $O(m^2)$ time.

Proposition 1. *Given m steps organized into a single job, the time-optimal AI strategy can be calculated in time $O(m^2)$ via dynamic programming.*

Proof. We first note the following recursive formulation of the optimization problem. For all $k \leq m$, let $C[k]$ denote the minimum time needed to complete a hypothetical job that only includes steps 1 through k. Note that, because of the way we defined $C[k]$, task k cannot be automated in any minimum-time solution that determines $C[k]$; it can only be augmented or performed manually. Note also that $C[m]$ is the time cost of our desired optimal solution.

We now show how to calculate $C[k]$ recursively. As a base case we have $C[0] = 0$, as the empty set of steps requires no time to complete. For $k \geq 1$, $C[k]$ is the lesser of

$$C[k-1] + t_k^M$$

and

$$\min_{\ell<k}\left\{C[\ell] + \frac{t_k^A}{\prod_{i=\ell+1}^{k} q_i}\right\}.$$

In other words, we either complete step k manually (in which case we separately optimize over steps 1 through $k-1$) or we augment step k. In the latter case, we then optimize over the length of the AI chain that ends with the newly-augmented step k. This could be a singleton chain with no automation (corresponding to $\ell = k-1$), or a longer chain that automates one or more steps before step k (corresponding to a choice of $\ell < k-1$). For any such choice of ℓ, we then separately optimize the production strategy for tasks 1 through ℓ.

Using this formulation, given the values $(C[0], \ldots, C[k-1])$, we can calculate $C[k]$ in time $O(k)$ by considering each potential choice of ℓ. Doing so for each $k = 1, 2, \ldots, m$ yields the value of $C[m]$ (and the corresponding optimal AI strategy) in total time $O(m^2)$. □

4.2 Warm-up to Long-Term Optimization: Job Design without AI

We now move to a broader optimization problem in which the firm can reorganize job requirements and the assignment of tasks to workers. As a warm-up to the full joint optimization of jobs and AI deployment, we first show how to optimize the design of jobs for a fixed task structure. We can interpret this as a labor optimization problem without AI deployment, where each step has only a single (i.e., manual) mode of completion and hence the task structure is fully determined.

Proposition 2. *Given a fixed sequence of tasks $\mathcal{T} = (T_1, \ldots, T_n)$ with skill and time costs $c = (c_1, \ldots, c_n)$ and $t = (t_1, \ldots, t_n)$, the cost-minimizing job design $\mathcal{J}$ can be computed in time $O(n^2)$ by dynamic programming.*

Proof. We first note the following recursive formulation of the optimization problem. For all $k \leq n$, let $C[k]$ denote the minimum cost of a job design for tasks 1 through k, *including hand-off costs for task k*. Note then that $C[n]$ is the cost of the optimal job design for all tasks, recalling that the hand-off cost for the final job (i.e., the job that includes task n) is always 0.

We now show how to calculate $C[k]$ recursively. As a base case we have $C[0] = 0$. For $k \geq 1$ we have

$$C[k] = \min_{s<k} \left\{ C[s] + \left[\left(\sum_{i=s+1}^{k} c_i \right) \left(t_k^H + \sum_{i=s+1}^{k} t_i \right) \right] \right\}.$$

In other words, we optimize over the choice of the final job $\{s+1, \ldots, k\}$, accounting recursively for the optimal job design for remaining jobs. Note that we make implicit use of the assumption that the hand-off cost t_k^H depends on the final task T_k of this final job but is otherwise independent of the job design. Using this formulation, we can calculate each $C[k]$ in time $O(k)$ by considering each potential choice of s. Doing so for each $k = 1, 2, \ldots, n$ yields the value of $C[n]$ (and the corresponding optimal job design) in total time $O(n^2)$. □

4.3 Full Long-Term Optimization

We now consider full joint optimization of job design and AI deployment strategy, accounting for both time and skill costs of workers assigned to the resulting jobs. In other words, the firm's goal is to choose both the set of AI chains to implement—yielding a vector of skill and time costs—along with a job design for the resulting tasks.

4.3.1 Recursive Formulation of Optimization Problem

Here we present a recursive formulation of the firm's cost minimization problem. In Section 4.3.2 we propose an approach to calculate an approximation of the optimal solution via dynamic programming.

Consider a production process with m steps $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$. For $0 \leq i \leq m$, let $V(i)$ denote the minimum cost required to complete steps 1 through i using one or more jobs, including any hand-off costs to a subsequent worker. Note then that $V(m)$ is the solution to the desired optimization problem, recalling that the hand-off time for the final job will always be 0. Note also that it is implicitly assumed in the definition of $V(i)$ that it optimizes over job designs for steps 1 through i in which the final job terminates after the completion of step i.

It will also be helpful to consider optimal designs for steps 1 through i in which the final job doesn't necessarily terminate after the completion of step i, but rather continues onward to subsequent steps. To that end, we introduce an auxiliary function $W(i,\ c,\ t)$. This function denotes the minimum cost of completing steps 1 through i, as well as some additional (but unspecified) set of tasks that have already been assigned as a single job to a single worker (referred to as the "active worker"). It is assumed that the tasks assigned to the active worker have total time cost t and total skill cost c. The time cost t is assumed to already include any hand-off cost for this active worker (which recall depends only on the final step of the worker's final task). Crucially, the minimum is taken over all AI deployment strategies and job designs, including those that expand the number of tasks assigned to the active worker (e.g., by including step i in the active worker's job). In this sense, $W(i,\ c,\ t)$ captures the minimum cost across all feasible job designs and AI strategies that may potentially expand the active worker's task assignment.

Note that, for all $i \leq n$, we have

$$V(i) = W(i,\ 0,\ t_i^H).$$

This is because, in $V(i)$, all job designs must end with a job that completes at step i and hands off to the following worker. Consequently, the hand-off cost associated with step i must be incurred by the final worker. Equivalently, this can be thought of as assigning the active worker an additional task with zero skill requirement but a time cost equal to the hand-off cost of step i right after step i itself. This scenario is represented explicitly by $W(i,\ 0,\ t_i^H)$.

Also note that, for the base case $i = 0$, we have

$$W(0,\ c,\ t) = c\,t, \quad \text{for all } c, t.$$

This is because, if $i = 0$, then there are no further steps to add to the job of the active worker. The active worker's skill and time requirements are therefore determined entirely by the steps they have already been assigned.

We are now ready to describe a recursive formulation of the function $W(i,\ c,\ t)$ for $i \geq 1$. We emphasize that in the definition of $W(i,\ c,\ t)$, the active worker has not been assigned step i; we can think of step i as the highest-indexed step that has not yet been assigned to a worker.

Proposition 3 (Recursive Formulation of Job Design Problem)**.** *For each $i \geq 1$, the minimum cost function $W(i,\ c,\ t)$ satisfies the following recursive relation:*

$$W(i,\ c,\ t) = \min\Bigg\{ c\,t\ +\ V(i),$$
$$W(i-1,\ c + c_i^M,\ t + t_i^M),$$
$$\min_{r<i}\ W\left(r,\ c + c_i^A,\ t + \frac{t_i^A}{\prod_{s=r+1}^{i} q_s}\right)\Bigg\}.$$

The cost of the optimal joint AI strategy and job design for a given sequence of m steps is then $V(m) = W(m,\ 0,\ 0)$.

The recursive formulation in Proposition 3 evaluates the minimum cost among three distinct alternatives at each step:

- **Option (1):** The term $c\,t\ +\ V(i)$ corresponds to not adding any further steps to the job of the active worker. All steps 1 through i will be completed by other workers. The active worker's total cost is then $c\,t$, and $V(i)$ is the total cost of completing steps 1 through i (including hand-off costs to the active worker).

- **Option (2):** The term $W(i-1,\ c+c_i^M,\ t+t_i^M)$ corresponds to designating step i for manual completion, and adding the corresponding (singleton) task. In this case, the manual execution costs (c_i^M, t_i^M) of step i are added to the accumulated costs of tasks already assigned to the active worker, extending their job to include step i as well.

- **Option (3):** The term $\min_{r<i}\ W\left(r,\ c + c_i^A,\ t + \frac{t_i^A}{\prod_{s=r+1}^{i} q_s}\right)$ corresponds to creating an AI chain task for steps $r+1$ through i, with step i being augmented and steps $r+1$ through $i-1$

automated, and assigning the resulting task to the job of the active worker. Step i is thus chained with zero or more preceding (automated) steps, and the associated AI management costs are added to the active worker's skill and time costs. The index r corresponds to the highest-index step that is not added to this AI chain, which can be any value between 0 and $i-1$.

Note that in evaluating option (3) of the recursive step for $W(i,\ c,\ t)$, step i can only be augmented (rather than automated) as each step of the recursion adds a full AI chain to an active worker's job. AI strategies in which step i is automated are considered when performing task calculations $W(r, c, t)$ with $r > i$.

4.3.2 Calculating an Approximately Optimal Solution

Given our recursive formulation, we wish to use dynamic programming to solve for the jointly optimal job design and AI strategy by filling in the values of $W(i,\ c,\ t)$ for all i, c, and t. Since c and t take on continuous values, we will discretize all skill costs c and time costs t to appropriate powers of $(1+\epsilon)$, where $\epsilon > 0$ is an arbitrarily small error term. We obtain the following result.

Proposition 4. *Fix any sequence of m steps $\mathcal{S} = (s_1, \ldots, s_m)$ for which all skill costs, time costs, and hand-off costs lie in $[1/B, B]$ for some $B > 0$. Then for any $\epsilon > 0$, an approximately cost-minimizing pair of AI strategy $\mathcal{T}$ and job design $\mathcal{J}$ minimizing expression* (3) *to within a factor of $(1+\epsilon)$ can be computed in time $O(m^2\epsilon^{-2}\log^2(mB))$ by dynamic programming.*

Proof. The first step of proving Proposition 4 is to determine the range of powers of $(1+\epsilon)$ that we must consider in our discretization. Suppose that all manual and augmented skill and time costs, as well as all hand-off costs, lie in $[1/B, B]$ for some $B > 0$. In this case, note that any subsequence of steps could be completed at a total cost of at most $2mB^2$ by performing them all manually by different workers, incurring a time and hand-off cost of at most B each for a total time of $2B$, and a skill cost of at most B for a total cost of $2B^2$ for each of the m tasks. We therefore need not consider any solution containing a job with total cost greater than $2mB^2$. In particular, since any non-zero skill cost for any job is at least $1/B$, we need not consider any job design in which a job has total time cost greater than $2mB^3$. Furthermore, each job has total skill cost lying between $1/B$ and mB (as skill costs combine additively) and time cost at least $1/B$ (as each step's manual cost is at least $1/B$, and any AI chain's time cost is at least the management cost of its final step which is also at least $1/B$).

We conclude that when filling table $W(i, c, t)$, it suffices to consider skill costs c lying in the range $[1/B, mB]$ and time costs t lying in the range $[1/B, 2mB^3]$. There are $O(\epsilon^{-1}\log(mB))$ powers of $(1+\epsilon)$ in each of these ranges, so our table will have $O(m\epsilon^{-2}\log^2(mB))$ total entries. Each entry can be filled in time $O(m)$ by considering the recursive formulation in Proposition 3, for a total runtime of $O(m^2\epsilon^{-2}\log^2(mB))$.

Note that while the table formally describes only the cost of a solution and not the solution itself, one can also read off the corresponding task and job design as well. Indeed, by recording which of the options described in Proposition 3 achieves the minimum for each table entry $W(i, c, t)$, one can trace out the corresponding task and job designs starting with $V(m) = W(m, 0, 0)$. The pair of AI strategy $\mathcal{T}$ and job design $\mathcal{J}$ can therefore be computed in time $O(m^2\epsilon^{-2}\log(mB))$, the same as the time required to fill the table.

It remains to bound the error introduced by our discretization. Consider the recursive formulation in Proposition 3 and suppose that c and t are rounded down to the nearest power of $(1+\epsilon)$.

Since time and skill costs are multiplied together to calculate the total cost of a proposed job, this discretization introduces a multiplicative error of at most $(1+\epsilon)^2$ into our cost calculation for the first option when minimizing in the recursive definition of $W(i, c, t)$. For the other two options, note that we accumulate time and skill costs additively when chaining tasks together for a single worker. As c and t are rounded to a power of $(1+\epsilon)$, adding additional (accurate) skill and time costs and subsequently rounding maintains a multiplicative error of at most $(1+\epsilon)$ on the total time and skill costs.

We conclude that if we fill in our table for all $W(i,\ c,\ t)$ where $i \leq m$ and c and t are discretized into powers of $(1+\epsilon)$, rounding down to nearest values of $(1+\epsilon)$ on recursive calls into the table, we obtain a $1+O(\epsilon)$ approximation to the optimal solution. An appropriate change of variables (scaling ϵ by a constant) yields a $(1+\epsilon)$ approximation factor in total runtime $O(m^2\epsilon^{-2}\log^2(mB))$, as claimed in Proposition 4. □

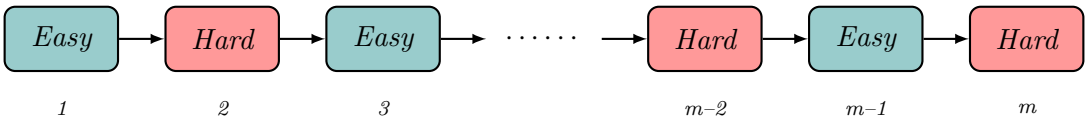
5 Discussion

5.1 Job-level AI Exposure and the Fragmentation Index

For a single step in isolation, the decision whether or not to deploy AI augmentation depends simply on whether the manual execution cost exceeds the AI management cost. That is, the optimal choice of whether to use AI assistance depends only on the AI exposure of the step in question. For two or more steps, however, the optimal deployment of AI depends on more than the exposure of each step in isolation. This can arise because of the benefits of automating multiple steps together in an AI chain. Even if one step can be completed more effectively through manual work, it may be preferable to automate it as part of a larger collection of steps that can be jointly delegated to AI. Whether this occurs in the optimal AI strategy depends on the relationship between other nearby steps in the production process.

Returning to the short-run optimization problem described in (4) and Section 4.1, we can interpret job-level AI exposure as the extent to which an optimal AI strategy employs AI automation and augmentation. Intuitively, a run of *consecutive* steps in the production process that an AI can perform effectively is a natural candidate for an AI chain. A job that contains such runs of consecutive steps is therefore likely to benefit most from AI automation. On the other hand, a job for which AI-friendly steps are separated by intermediate steps that an AI is likely to fail is less exposed to large-scale automation, even though its task-level exposure to AI may appear high.

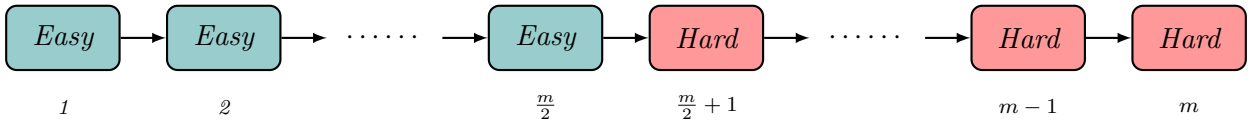
Example 1. *Consider a job consisting of m steps, where m is even. Each step takes time 5 to complete manually and has an AI management time of 1, but the steps vary in how difficult they are for an AI to complete: odd-numbered steps can be successfully completed by an AI with probability 1, whereas even-numbered steps will be successfully completed with probability only 0.1, as shown below:*



In this scenario, it is suboptimal to attempt to use AI on any of the even-numbered steps, even as

part of a chain. The optimal AI deployment strategy is to employ AI-augmentation on the odd-numbered steps (for a cost of 1 *each) and perform even-numbered steps manually (for a cost of* 5 *each), resulting in an overall time cost of* $3m$*.*

Example 2. *Next suppose that easy and hard steps are not interleaved, but rather the first* $m/2$ *steps can each be completed by AI with probability* 1 *and the last* $m/2$ *steps can each be completed with probability* 0.1*, as follows:*



In this case, the optimal AI strategy chains together the first $m/2$ *steps into a single automated AI chain, for a combined cost of* 1*. The remaining* $m/2$ *steps are then performed manually. This results in an overall time cost of* $1+5m/2$*, which is strictly less than* $3m$ *as long as there are four or more steps.*

To make this intuition more precise, let us return to the short-run optimization problem described in (4) and Section 4.1. Recall that in this short-run problem we are fixing a single job assigned to a worker with a fixed wage, so our focus is on optimizing the time cost of production. Consider the special case where AI management costs are normalized: $t_i^A = 1$ for all i. That is, each step of production requires the same amount of time to prompt and manage one attempt by an AI process. Under this assumption, we will define a measure of the dispersion of AI-exposed tasks in a production process, which we refer to as the *fragmentation index* of a job. We then show that the fragmentation index of a job approximates (up to constant factors) the time cost of an optimal AI strategy for that job. In other words, jobs for which the fragmentation index is high will tend to yield less benefit from AI automation in the short-term where worker wages and job boundaries are fixed. Notably, this intuition makes heavy use of the short-run assumption that worker wages are fixed and independent of task skill requirements; we discuss examples where this intuition fails when describing long-run effects in Section 5.2.

Intuitively, the fragmentation index is motivated by a hypothetical scenario where a prophet can "see the future" to determine which tasks would be completed successfully by AI on its first attempt and which would not. If many tasks in sequence will all complete successfully on the first try, these are a good candidate for an AI chain. The prophet might therefore chain together all contiguous sequences of tasks that would all succeed, and perform all other tasks manually. The fragmentation index is then the expected cost of this strategy, under the assumption that the prophet's foresight is correct. Notably, this AI strategy isn't necessarily optimal even with the benefit of foresight: a long AI chain with a good probability of success might be optimal even if the prophet knows that it will fail on the first attempt. However, what we show is that its cost approximates the cost of the optimal strategy even without foresight.

To define the fragmentation index more formally, consider a random process in which each step s_i succeeds independently with probability q_i; any task that does not succeed is said to fail. Write F for the set of steps that fail, and $\mathcal{C} = \{C_1, \ldots, C_k\}$ for the random variable representing the collection of maximal connected components of non-failed steps. The weight of each $C_j \in \mathcal{C}$ is defined to be $\omega(C_j) = \min\{1, \sum_{i \in C_j} t_i^M\}$. That is, each C_j has weight 1 unless the sum of the

manual time costs for each of its steps is less than 1 (due to our assumption that AI management costs are normalized to 1).

Given a realization of $\mathcal{C}$ and F, we define the realized fragmentation to be

$$\sum_{i \in F} \min \left\{ t_i^M, \frac{t_i^A}{q_i} \right\} + \sum_{C_j \in \mathcal{C}} \omega(C_j). \tag{5}$$

The *fragmentation index* is defined to be the expected value of the realized fragmentation.

Intuitively, we expect the fragmentation index to be lower when highly automatable steps (that is, those for which the AI are likely to succeed) are clustered together, since a large cluster of highly automatable steps are more likely to realize as a single connected component when failures are realized.

We will show that the fragmentation index is within a constant factor of the cost of the optimal (short-term) AI strategy for a given fixed job's step sequence.

Proposition 5. *Fix a single job with a sequence of m steps $\mathcal{S} = \{s_1, \ldots, s_m\}$, each with $t_i^A = 1$. Let FI denote its fragmentation index and let OPT denote the minimum time cost over all AI strategies. Then $\frac{1}{8}OPT \leq FI \leq \frac{5}{4}OPT$. If we further assume $t_i^M \geq 1$ for all i, then $\frac{1}{4}OPT \leq FI \leq \frac{5}{4}OPT$.*

We prove Proposition 5 in Appendix A. The key take-away from Proposition 5 is that jobs with high fragmentation index—i.e., those for which the expected number of *consecutive* successful step executions by an AI is low—will tend to have higher time costs even under optimal use of AI chaining. A key driver of efficiency gains via AI automation is therefore not simply the expected number of steps that can be automated effectively, but the number of *adjacent* steps in the production process that have high exposure to AI.

An implication of our proof of Proposition 5 is a natural and approximately optimal greedy algorithm for constructing AI strategies. The algorithm groups tasks into chains as long as the probability of success is sufficiently high; if the success probability falls too low then the chain is terminated and a new chain is started (with chains of length 1 converted to manual execution when appropriate).

5.2 Impact of AI Deployment on Worker Skill and Specialization

While the fragmentation index captures the short-run impact of AI deployment on task completion time, in the long run AI strategy also alters the skill requirements of workers and the degree of specialization within the workforce. This can dampen or even reverse the short-run intuition that gains from AI derive primarily from time savings, as the following example shows.

Example 3. *Consider a job consisting of a single step. This step has manual time and skill requirements $(t^M, c^M) = (5, 5)$. Suppose that, under AI augmentation, this step has an AI management time of 1 (per attempt), an AI management skill requirement 1, and probability $1/8$ of successful completion. That is, $(t^A, c^A) = (1, 1)$ and $q = 1/8$. Despite the task taking longer to complete with AI (due to the low likelihood of success on any individual attempt), the reduced skill requirement of managing the AI process means that it is firm-optimal to employ AI augmentation.*

The impact of AI on worker skill reflects two common narratives about the impact of AI on work: first, that AI can automate mundane or repetitive tasks and allow high-skill workers to concentrate

more of their time on meaningful high-skill tasks; second, that AI could lead to deskilling as high-skilled labor for manual task completion is replaced with low-skill AI management. Our framework unifies these two perspectives by endogenizing (a) the nature (i.e., time and skill requirements) of work to be automated and (b) how the AI strategy impacts requisite worker skill. If replacing a given sequence of production steps in given job with an AI chain is beneficial, in the long term, then either the total time needed to complete the steps is shortened, the total skill needed to manage the AI automation is reduced relative to completing the steps manually, or both. For an example of the latter, a step with low skill requirement but high time requirement (such as Step 2 in Figure 3) may be optimally augmented by AI even when doing so increases the skill needed for AI management, if the time savings are substantial enough. In this case, the skill required to complete the same job might increase: AI augmentation can be viewed as complementing worker capabilities. Alternatively, it may also be beneficial to employ AI on production steps with high skill requirements (such as Step 1 in Figure 3) even if this results in an AI-assisted task that takes longer to complete, as long as it requires substantially less worker skill to manage the AI. This can ultimately lead to a deskilling of the labor force assigned to the job as AI capital substitutes for high-skilled human work.

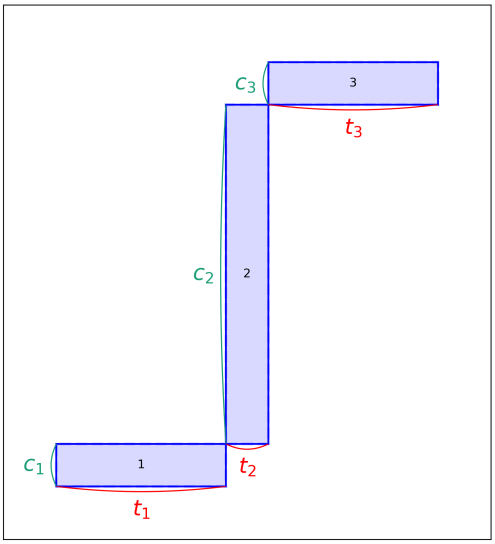
This discussion has so far focused on impacts within a single job. Our framework likewise captures how AI deployment strategy shapes job design and, in particular, patterns of worker specialization across tasks. Consider again the example from Figure 3, which highlights how hand-off costs and the structure of what we refer to as *tent-pole tasks* can influence how production steps are bundled into specialized jobs. A tent-pole task is a short but high-skill task that sits immediately next to time-consuming, low-skill ones.[12] The second task in Figure 4 is an example of a tent-pole task.

As AI deployment strategy influences the time and skill profile of tasks, the incentive to specialize workers likewise changes. To predict the direction of this effect, one hypothesis is that AI deployment will tend to have a normalizing effect on tasks, reverting both skill requirements and time requirements toward the mean and making tasks more "square-like" (in terms of our geometric interpretation of job design costs). If so, and if hand-off time costs remain unaffected, then (roughly speaking) tent-pole inefficiencies due to grouping tasks together into the same job would be reduced. This suggests that AI deployment may reduce worker specialization.

Example 4. *Consider a two-step production process, with manual skill and time costs* $(c_1^M, t_1^M) = (2, 4)$ *and* $(c_2^M, t_2^M) = (4, 2)$ *and hand-off cost* $t_1^H = 5$*. That is, the first step is low-skill but time-intensive and the second step is high-skill but can be completed quickly. In this example, combining both steps into a single job (at a labor cost of* $(2+4)(4+2) = 36$*) is strictly worse than separating into two specialized jobs with a hand-off (at a total cost of* $(2)(4+5) + (4)(2) = 26$*). However, if each step of production could be augmented separately via AI to yield effective skill and time costs of* $(2, 2)$ *for each, then the optimal design combines both augmented steps into a single job at a total cost of* $(2+2)(2+2) = 16$*, which is better than separating into two distinct jobs (incurring cost* $(2)(2+5) + (2)(2) = 18$*).*

[12] This juxtaposition creates a classic friction: if a single highly skilled worker performs all adjacent tasks, a substantial share of their time is spent on low-skill work, whereas assigning the surrounding low-skill tasks to different workers raises hand-off costs when work passes between them. Historically, this tension has been an important driver of the division of labor. As AI automation alters the time and skill profiles of individual tasks in the production sequence, it changes exactly this trade-off, shifting when it is efficient to have a single worker perform a cluster of tasks versus when specialization across workers remains optimal.

Figure 4: Illustration of Tent-Pole Tasks



Notes: This figure illustrates tent-pole tasks. The width and height of each rectangle specify the time (t) and human capital (c) requirements of the task, respectively. Tasks 1 and 3 have a high time cost and low human capital requirement (wide and short) whereas Task 2, which is the tent-pole task, exhibits a low time cost and high human capital requirement (narrow and tall). The hand-off time costs (t^H) are assumed to be zero for simplicity.

It is also technically possible in our model for AI to increase the amount of specialization in an optimal job design. This could happen if AI-augmented steps have higher skill requirements than the corresponding manual steps, leading to an increased need for high-skill workers. Even if AI management is assumed to require no more skill than manual completion, an increased deployment of AI can lead to more responsibilities being combined into a single job, leading to a higher worker skill requirement, as the following example shows.

Example 5. *Consider a different two-step production process, with manual skill and time costs* $(c_1^M, t_1^M) = (c_2^M, t_2^M) = (3, 3)$ *and hand-off cost* $t_1^H = 5$*. The optimal job design separates these into two separate jobs at a total cost of* $(3)(3+5)+(3)(3) = 33$*, with each job requiring a worker of skill* 3*. Suppose AI augmentation can allow each step to be completed with an AI management skill of* 2*, a management time of* $1/4$*, and an AI success probability of* $1/8$*, for a total expected execution time of* $(1/4) \times (1/8)^{-1} = 2$*. In this case, the optimal design employs AI augmentation in each step and combines the two steps into a single job, resulting in a total cost of* $(2+2)(2+2) = 16$ *and requiring a worker of skill* 4*. Note that this is preferable to combining the two steps into a single AI chain, which would result in a total cost of* $(2)((1/4) \times (1/8)^{-1} \times (1/8)^{-1}) = 32$*.*

5.3 Non-Linear Impacts of AI Improvements

Improvements to AI technology naturally lead to improvements in the overall cost of production, but these effects need not be linear. The full economic impact of a disruptive general-purpose technology can take significant time and investment to materialize, with notable historical examples

ranging from the steam engine to electricity to computers. In each case, a new technology enables modest short-term improvements to existing production processes, but the full impact is not felt until large-scale reorganization of production is enabled. This drives a potentially non-monotone marginal value from technology improvements, where major improvements are only unlocked after surpassing a threshold of capability at which they become feasible.

Our framework captures such non-monotonicities in the marginal value of technological improvements. This arises in our model via optimization over different AI deployment and job design strategies. One can think of parameter $\alpha \in (0, 1]$ in our model as metric of the quality of a general purpose AI technology, taken to be the probability that a task of normalized difficulty 1 is completed successfully. We can then model general technology improvements as increasing the value of α. For a given *fixed* AI strategy (described by a task sequence $\mathcal{T}$) and job design (described by job sequence $\mathcal{J}$), the total cost can be expressed as a polynomial in $(1/\alpha)$. As a result, the cost of $\mathcal{T}$ and $\mathcal{J}$ shifts in a continuous and smooth manner as α increases, with monotone marginal gains to technology improvements. However, the optimal choice of design (expressed as optimization problem (3)) involves taking the minimum-cost solution over many possible AI strategies and job designs. AI strategies with a higher degree of AI chaining will naturally involve higher powers of $1/\alpha$ in their cost expressions, meaning that they are more sensitive to changes in α and become preferable only at higher values of α (i.e., as AI becomes more effective as a tool). This naturally leads to scenarios where the productivity improvements of AI integration are modest at low levels of quality but can grow sharply past a certain inflection point, as we describe in the following example illustrated in Figure 5 below.

Example 6. *Consider a production process with two steps. The first step is short but high-skill and difficult for AI tools to perform correctly. Specifically, its manual skill cost is* $c_1^M = 5$*, its manual time cost is* $t_1^M = 1$*, and its hand-off time is* 1*. Its AI difficulty score,* d_1*, is* 6*, meaning that an AI can successfully complete the step with probability* $q_1 = \alpha^6$*. The AI management skill and time requirements are the same as the manual execution requirements:* $(c_1^A, t_1^A) = (5, 1)$*.*

The second step is low-skill and time-consuming to execute manually, but easy for AI. It has $(c_2^M, t_2^M) = (2, 4)$ *and AI difficulty score* 1*. The skill needed to manage the AI is also* 2*, but the time needed to manage the AI is reduced to* 1*:* $(c_2^A, t_2^A) = (2, 1)$*.*

Consider the optimal AI strategy and job design for this example as a function of α*, the general AI quality metric. The costs of different AI strategies are plotted in Panel (a) of Figure 5. For* $\alpha < 0.25$*, it is optimal to complete each step manually as a separate task, and to separate the two tasks into separate specialized jobs. This incurs a total cost of* $(5)(1 + 1) + (2)(4) = 18$*. As this strategy does not employ AI, the cost is not affected by improvements to* α *in this range, which is evident in Panel (b) of the figure.*

For $\alpha \in (0.25, 0.77)$*, the optimal AI strategy changes: it is better to use AI augmentation for the second step, reducing that step's total completion cost from* $(2)(4)$ *to* $(2)(1/\alpha)$*. As* α *increases, the total execution cost drops gradually from* 18 *(at* $\alpha = 0.25$*) to* $10 + 2/\alpha \approx 12.5$ *(at* $\alpha \approx 0.77$*). This gradual improvement is visible as a modest slope in Panel (b).*

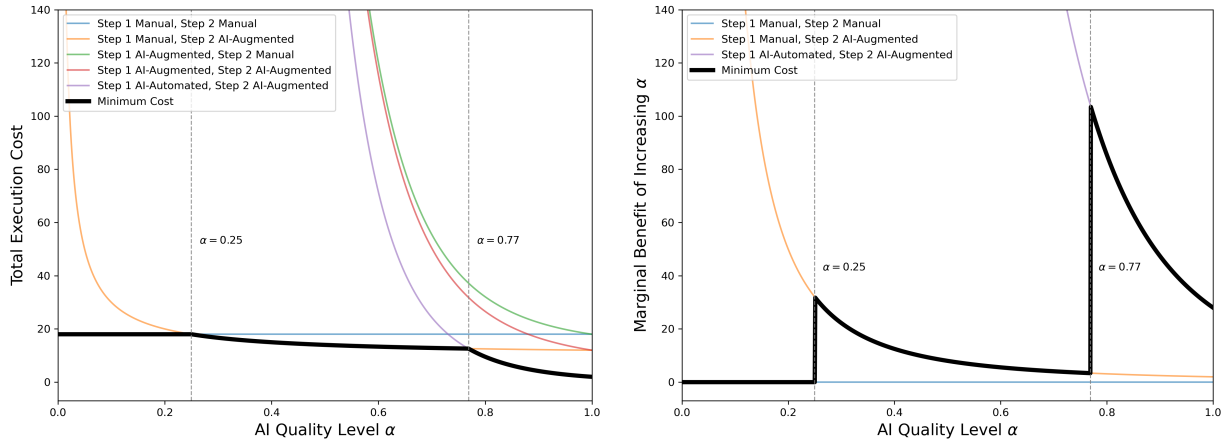
However, once α *grows larger than* 0.77*, the optimal AI strategy and job design change more dramatically. At this point, it is better to combine both steps into a single AI chain, automating step* 1*, and to assign the resulting aggregate task to a single low-skill worker. The completion cost of this strategy is* $(2)(1/\alpha^7)$*. This cost starts at approximately* 12.5 *at* $\alpha = 0.77$ *but drops sharply, eventually converging to a total cost of* 2 *at* $\alpha = 1$*. This sharp drop indicates substantial marginal benefits from incremental improvements in AI quality in this region, illustrated by the pronounced*

jump in the slope in Panel (b) at $\alpha \approx 0.77$.

Figure 5: Cost Analysis of AI Strategies in Example 6

(a) Total Cost by AI Strategy

(b) Marginal Benefit of Improving AI Quality



Notes: Panel (a) shows total execution costs under various AI strategies as AI quality α improves. Panel (b) illustrates the marginal reduction in optimal costs associated with increased AI quality. Dashed vertical lines indicate thresholds where optimal AI deployment strategies shift.

6 Macro-level Production Function

So far, the model has focused on individual firm decision-making: how a firm allocates production steps among automation, augmentation, and manual execution, and how its micro-level choices of task assignment and job design determine internal productivity and costs. Ultimately, however, our interest extends beyond the firm to the broader economy. Improvements in AI quality not only alter how individual firms organize production but also affect the composition of inputs used across the economy and the aggregate relationship between labor, capital, and output. Yet the firm-level optimization framework developed above prevents direct analysis of these aggregate effects and also makes our approach distinct from the large literature in which production is modeled directly at the task level and then aggregated to the economy level (e.g., Acemoglu et al. (2022, 2024); Acemoglu (2025)). To bridge this gap, this section examines whether the micro-level cost-minimization problem of firms can be aggregated into well-behaved firm- and economy-level production functions. In particular, we show that the detailed, many-input Leontief representation of production at the task level can be reduced to a more compact formulation with only a small number of effective inputs, and that aggregating across heterogeneous firms (in terms of AI quality level) yields a macro-level production function with a CES form. This provides a tractable link between micro-level organization decisions and macroeconomic implications of improving AI quality.

We begin by showing how a firm's micro-level decisions can be explicitly represented by a Leontief production function at the task level. Next, we demonstrate how, for the purposes of analyzing labor allocation within a firm, this detailed task-level production function can be simplified into a

Leontief production function with just two aggregate inputs: skill-adjusted AI management labor and skill-adjusted manual labor (we will describe what we mean by "skill-adjusted" below). Finally, we explore how the production functions of numerous individual firms, which differ only in how they can effectively put the same AI technology into use, can be aggregated into a single, economy-wide CES production function. In this macro-level representation, the inputs are economy-wide aggregates of the corresponding micro-level inputs from individual firms, thereby facilitating clearer insights into the overall labor allocation in the economy.

Before the aggregation analysis, let us first describe the setup of production. A firm uses two production inputs: (1) skill-adjusted AI management labor (to complete AI-assisted tasks), and (2) skill-adjusted manual labor (to perform manual tasks). These two types of labor demand different compensations: executing an AI-assisted task with skill and time cost of one demands w_A, while completing a manual task with similar skill and time requirements commands a compensation of w_M. We call w_A and w_M the *base wage rates* for AI management and manual labor, respectively.

Firms use these two inputs to complete tasks. Recall that the skill and time requirements of tasks are determined by the firm's AI strategy $\mathcal{T}$, while the aggregate skill requirements of jobs depend jointly on the AI strategy and job design $\mathcal{J}$. Specifically, the per unit of time compensation of a worker employed to do tasks of a job is determined by all tasks in that job. Consider Job 1 in Figure 2 for example. The worker assigned to perform Job 1 is required to obtain not only the skill required for manual Task 1, c_1^M, but must also possess the required skill for AI-assisted Task 2, c_2^A, as the worker's total compensation per unit of time is determined at the job level, and equals $c_1^M + c_2^A$. Therefore, the more tasks firm includes in a job, the higher required compensation for every task in that job.

In order to produce, the firm hires manual labor to perform manual tasks and AI management labor for AI-assisted tasks. Therefore, the total compensation of a job will be a weighted average of skill costs of the job's tasks, with weights being the base wage rates of each type of labor. Specifically, the compensation for job J will be:

$$w_M \left(\sum_{T_b^M \in J} c_b^M \right) + w_A \left(\sum_{T_b^A \in J} c_b^A \right), \tag{6}$$

which is composed of the sum of contributions of skill costs from its manual tasks denoted by set T_b^M (each weighted by w_M) and its AI-assisted tasks denoted by set T_b^A (each weighted by w_A).[13]

Next, we define skill-adjusted labor, which is the smallest unit of work in our analysis. Let $J(b)$ denote the job to which task b belongs, and $E(b)$ be the mode of execution of task b (i.e., $E(b) = M$ if b is a manual task, or $E(b) = A$ if b is an AI chain). Define the skill-adjusted time requirement of task b as

$$\tau_b = \frac{w_M \left(\sum_{T_\ell^M \in J(b)} c_\ell^M \right) + w_A \left(\sum_{T_\ell^A \in J(b)} c_\ell^A \right)}{w_{E(b)}} t_b.$$

Note that the fraction appearing behind t_b represents an effective skill adjustment factor, as the numerator captures the total compensation of the job task b belongs to, while the denominator

[13] This formulation is essentially equivalent to the original definition of wage in (1), as pointed out in Footnote 9. If we multiply each skill cost in (6) by its respective base wage rate and redefine the task's skill cost as the resulting product, we arrive at the original formulation given by (1). The formulation in (6) allows tasks to contribute differently to the job's wage depending on their mode of execution and type of labor that need to perform them.

normalizes by the base wage rate for the specific mode of execution of task b. The resulting fraction thus has units of skill intensity. This characterization becomes useful later as it allows expressing the (expected) wage bill paid for task b simply as $w_A\,\tau_b\,\alpha^{-d_b}$ if it is AI-assisted or as $w_M\,\tau_b$ if it is executed manually.

6.1 Within-Firm Aggregation: Leontief to Leontief

We now consider the firm's production. To produce output, each firm must complete requirements of all m production steps $s_1, \ldots, s_m$. This naturally results in a Leontief production structure with multiple inputs.[14] Although firms choose whether to execute each step manually or by the help of AI, we show that analyzing labor allocation can be simplified. Specifically, production function of the firm can be represented by a Leontief function with only two firm-level aggregate inputs: AI management labor and manual labor.

Suppose the firm solves the long-run cost minimization problem (3) for a given AI quality level α, and obtains the optimal AI strategy $\mathcal{T}$ and job design $\mathcal{J}$. Let $|\mathcal{T}| = n$, implying the original m production steps are organized into n distinct tasks, and $|\mathcal{J}| = p$, indicating these n tasks are grouped into p jobs. Given the fixed AI strategy, the execution mode for each production step is predetermined. This lets us consider and work directly with tasks rather than individual steps. Furthermore, with the job design held fixed, each firm takes the job boundaries and the total skill requirements within jobs as given.

We assume hand-offs are performed manually by the worker who completes the final task of each job. This allows us to simplify the analysis in two ways: (1) Since job design is fixed, we know exactly which steps occur at the job boundaries and thus incur hand-off time costs. Therefore, hand-off of job J_j can be treated as a standalone, human-executed task with time requirement $t^H(J_j)$ and skill requirement equal to the total skill requirement of job J_j.[15] (2) The order of tasks—including the newly defined hand-off tasks—can be rearranged within the production sequence. Specifically, we relabel tasks such that tasks 1 to k are AI-assisted and thus require AI management labor, while tasks $k+1$ to n, along with the hand-off tasks $n+1$ to $n+p-1$, are executed manually and require manual labor.[16]

The firm's task-level production can be expressed in the following Leontief form:

$$x = \min\left\{ \frac{l_1}{\tau_1^A\,\alpha^{-d_1}}, \cdots, \frac{l_k}{\tau_k^A\,\alpha^{-d_k}}, \frac{l_{k+1}}{\tau_{k+1}^M}, \cdots, \frac{l_n}{\tau_n^M}, \frac{l_{n+1}}{\tau^H(J_1)}, \cdots, \frac{l_{n+p-1}}{\tau^H(J_{p-1})} \right\}. \tag{7}$$

The x on the left-hand side represents the firm's output while l_b is the amount of labor assigned to task b. All other variables remain as previously defined.

Since the AI strategy and job design are fixed, the required amount of skill-adjusted time to spend on each task in the denominators of (7) are fixed, and the firm takes them as given. In equilibrium, allocation of labor to tasks satisfies

$$x = \frac{l_1}{\tau_1^A\,\alpha^{-d_1}} = \cdots = \frac{l_k}{\tau_k^A\,\alpha^{-d_k}} = \frac{l_{k+1}}{\tau_{k+1}^M} = \cdots = \frac{l_n}{\tau_n^M} = \frac{l_{n+1}}{\tau^H(J_1)} = \cdots = \frac{l_{n+p-1}}{\tau^H(J_{p-1})}.$$

[14] Different AI deployment strategies and job designs lead to different input proportions but share the same fundamental Leontief form.

[15] Denote the skill-adjusted time cost of hand-off task of job J_j with $\tau^H(J_j)$.

[16] Recall that the final job p requires no hand-off by definition. The last hand-off thus occurs at the end of job $p-1$.

Notice that the ratio of labor allocated to any two tasks a and b is independent of output level and wage rates (regardless of whether tasks a and b are executed manually or by the help of AI), and solely depends on their relative skill-adjusted time requirements, which are fixed given $\mathcal{T}$ and $\mathcal{J}$. The fixed ratio of labor inputs implies that the rate of substitution between any pair of tasks is independent of the labor allocated to other tasks:

$$\frac{\partial}{\partial l_z}\left(\frac{\frac{\Delta x}{\Delta l_a}}{\frac{\Delta x}{\Delta l_b}}\right) = 0, \qquad \forall z \neq a, b. \tag{8}$$

A necessary and sufficient condition to aggregate the firm's production function from a Leontief with one input per task into a Leontief with (firm-level) aggregate AI management labor and manual labor is that the rate of substitution between any two tasks within the same aggregate input type is independent of all tasks in the other aggregate input type (Leontief, 1947; Fisher, 1965; Felipe and Fisher, 2003). In other words, the relative amount of labor allocated to any two human-executed (AI-managed) tasks should depend exclusively on tasks within the human-executed (AI-managed) group.

The condition expressed in (8) satisfies the aggregation requirement above.[17] Thus, the firm's production function can be represented as:

$$x = \min\left\{\frac{\bar{\alpha}\, l_A}{\tau_A}, \frac{l_M}{\tau_M}\right\}. \tag{9}$$

Here, l_A and l_M represent respectively the firm-level aggregate AI management labor and manual labor. Moreover, the aggregate skill-adjusted AI and manual time requirements, τ_A and τ_M, are defined as the sum of skill-adjusted time requirements of tasks in their respective groups:

$$\tau_A = \sum_{b=1}^{k} \tau_b^A, \tag{10}$$

$$\tau_M = \sum_{j=1}^{p-1} \tau^H(J_j) + \sum_{b=k+1}^{n} \tau_b^M. \tag{11}$$

Finally, $\bar{\alpha}$ is the firm's effective AI quality level defined as

$$\bar{\alpha} = \frac{\sum_{b=1}^{k} \tau_b^A}{\sum_{b=1}^{k} \tau_b^A \alpha^{-d_b}}, \tag{12}$$

so that the following relationship holds:

$$\frac{\tau_A}{\bar{\alpha}} = \sum_{b=1}^{k} \tau_b^A \alpha^{-d_b}.$$

[17] In fact, this condition is stricter than necessary. It not only guarantees labor independence across aggregated input types, but also imposes a stronger condition: that the rate of substitution between tasks within the same aggregate input type is independent even of other tasks within the same aggregate type.

In short, (9) allows us to analyze labor allocation using the simpler two-input Leontief production function for a given AI strategy $\mathcal{T}$ and job design $\mathcal{J}$. This firm-level aggregate Leontief function implies the equilibrium condition:

$$x = \frac{\bar{\alpha}\, l_A}{\tau_A} = \frac{l_M}{\tau_M}. \tag{13}$$

The representation in (9) helps us gain clearer insights into the allocation between manual and AI management labor at the firm level by removing the need to deal with the complex, many-input production function (7) at the level of individual tasks.

6.2 Cross-Firm Aggregation: Leontief to CES

Next, we analyze whether Leontief production functions of many firms can be collectively represented by a single economy-wide production function, where each input in the macro function aggregates the corresponding inputs from the micro functions. To do so, we draw on the extensive literature on aggregation theorems in production theory, which establishes conditions for the existence of an aggregate production function based on individual firms' production functions. The central idea is to introduce an appropriate form of heterogeneity across firms, allowing micro-level production functions to be smoothed into a well-behaved aggregate function (see Sato (1975), Chapters 2 and 4, for existence conditions.).[18] While most existence results in this literature do not yield a closed-form functional representation, we show here that under certain conditions on firm heterogeneity, firm-level Leontief production functions can be aggregated into an economy-wide CES production function.

Consider a unit mass of firms in the economy, each producing output with AI management labor, human labor, and capital. We assume capital is a fixed input, so firms make labor decisions conditional on their given capital stock. Moreover, we impose a Leontief structure between capital and labor inputs, normalizing the production process so that exactly one unit of capital is required per unit of output produced.[19] Firms do not know their individual effective AI quality level before production occurs; instead, they share a common prior belief regarding the distribution from which these quality levels are drawn.

Production unfolds in two stages. In the first stage, firms solve the long-run cost minimization problem (3), committing to an AI deployment strategy $\mathcal{T}$ and a job design $\mathcal{J}$ based on their *expected* AI quality levels. Since firms hold the same expectations about the distribution of effective AI quality, they choose identical AI strategies and job designs. In the second stage, firms learn their realized effective AI quality levels, hire the required labor and capital, and begin production.[20] The resulting Leontief production function, determined by the previously selected AI strategy, job design, and the firm's realized effective AI quality, takes a form similar to (9).

This two-stage structure mirrors realistic firm behavior. Firms initially decide how to structure operations and post job vacancies. Only after making these structural commitments do they

[18] For instance, Houthakker (1955) studies aggregation from micro Leontief functions to a macro Cobb-Douglas function; Levhari (1968) investigates aggregation from micro Leontief functions to a macro CES function; and Sato (1969) explores aggregation from micro CES to macro CES functions. See Baqaee and Farhi (2019) for a broader formulation of this problem.

[19] This normalization can be interpreted as assuming identical capital productivity across firms. Since we introduce firm-level heterogeneity through variations in AI quality, we abstract away from additional sources of heterogeneity across firms.

[20] As production depends on the realized effective AI quality level, $\bar{\alpha}$ also determines firms' sizes.

hire workers and acquire machines to begin production. Once hiring occurs, firms discover the actual productivity levels of inputs, which necessitates operating under their previously chosen organizational structures.

In what follows, we demonstrate that the heterogeneity in effective AI quality level can be structured so that micro-level Leontief production functions (9) aggregate into a macro-level CES production function, in which the CES inputs are aggregates of the micro-level inputs (Levhari, 1968). Suppose specifically that the macro-level production function takes the form:

$$X = (\theta_A A^\rho + \theta_M M^\rho + (1 - \theta_A - \theta_M) K^\rho)^{\frac{1}{\rho}}, \tag{14}$$

where X represents the (economy-wide) aggregate output, A denotes aggregate AI management labor, M denotes aggregate manual labor, K is aggregate capital, and parameters θ_A and θ_M represent the weights of corresponding inputs. Since the measure of firms in the economy is normalized to 1, aggregate capital is also normalized to 1, making the third term in (14) effectively constant at $1 - \theta_A - \theta_M$.

To link the macro production function above to the micro production functions, we must relate aggregated firm-level inputs l_A and l_M from (9) to their corresponding economy-wide aggregates A and M in (14). This requires either determining the macro production function parameters from the distribution of heterogeneity, or specifying the form of heterogeneity given a set of macro production function parameters. We adopt the latter approach and derive the output probability density function in terms of effective AI quality, denoted by $\phi(\bar{\alpha})$, as a function of θ_A, θ_M, ρ. Throughout, we assume that $\rho < 0$ (which implies elasticity of substitution $\sigma < 1$), indicating that macro-level production exhibits some degree of complementarity between aggregate inputs.

Normalize the output price to $p = 1$, so that w_A and w_M can be interpreted as real wage rates for a unit of skill-adjusted AI management and manual labor, respectively. A firm produces only if it earns nonnegative profits:

$$w_A \, l_A + w_M \, l_M \leq x.$$

Substituting for l_M and x from (13) into the profitability condition yields:

$$w_A \, l_A + w_M \, \frac{\tau_M \, \bar{\alpha} \, l_A}{\tau_A} \leq \frac{\bar{\alpha} \, l_A}{\tau_A}.$$

Rearranging terms and canceling l_A gives the lower bound of firms' effective AI quality level:[21]

$$\bar{\alpha} \geq \frac{w_A \, \tau_A}{1 - w_M \, \tau_M}.$$

This lower bound implies that only firms with realized effective AI quality level above this threshold will produce in equilibrium and others exit the market. Moreover, observe from (12) that for $d_b \geq 0$ the upper bound of $\bar{\alpha}$ is 1 and is achieved when $\alpha \to 1^-$. Therefore:

$$\frac{w_A \, \tau_A}{1 - w_M \, \tau_M} \leq \bar{\alpha} \leq 1. \tag{15}$$

Let $\phi(\bar{\alpha})$ be the distribution of output across firms according to their effective AI quality level. A firm with effective AI quality $\bar{\alpha}$ thus produces output $x = \phi(\bar{\alpha})$ by definition. From equation (13), it directly follows that the AI management labor and manual labor used by this firm are:

[21] We assume the parameters w_A, w_M, τ_A, τ_M are such that $0 < (w_A \, \tau_A)/(1 - w_M \, \tau_M) < 1$.

$l_A = (\tau_A/\bar{\alpha})\,\phi(\bar{\alpha})$, and $l_M = \tau_M\,\phi(\bar{\alpha})$. Consequently, aggregate output X, aggregate AI management labor A, and aggregate manual labor M can be expressed as:

$$X = \int_{\frac{w_A\,\tau_A}{1-w_M\,\tau_M}}^{1} \phi(\bar{\alpha})\,d\bar{\alpha}, \tag{16}$$

$$A = \int_{\frac{w_A\,\tau_A}{1-w_M\,\tau_M}}^{1} \tau_A \frac{\phi(\bar{\alpha})}{\bar{\alpha}}\,d\bar{\alpha}, \tag{17}$$

$$M = \int_{\frac{w_A\,\tau_A}{1-w_M\,\tau_M}}^{1} \tau_M\,\phi(\bar{\alpha})\,d\bar{\alpha}. \tag{18}$$

Substituting the aggregated variables from the micro-level equations (16, 17, 18) into the aggregate production function (14), we obtain:

$$\int_{\frac{w_A\,\tau_A}{1-w_M\,\tau_M}}^{1} \phi(\bar{\alpha})\,d\bar{\alpha} = \left(\theta_A \left(\tau_A \int_{\frac{w_A\,\tau_A}{1-w_M\,\tau_M}}^{1} \frac{\phi(\bar{\alpha})}{\bar{\alpha}}\,d\bar{\alpha}\right)^{\rho} + \theta_M \left(\tau_M \int_{\frac{w_A\,\tau_A}{1-w_M\,\tau_M}}^{1} \phi(\bar{\alpha})\,d\bar{\alpha}\right)^{\rho} + (1-\theta_A-\theta_M)\right)^{\frac{1}{\rho}}. \tag{19}$$

Solving for $\phi(\bar{\alpha})$ in terms of θ_A, θ_M, ρ we get the following distribution for effective AI quality:

$$\phi(\bar{\alpha}) = \frac{(1-\theta_A-\theta_M)^{\frac{1}{\rho}}\left(1-\theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}}\left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}}}{1-\rho}(\bar{\alpha})^{\frac{1}{\rho-1}}\left[1-\theta_M\,\tau_M^{\rho}-\left(1-\theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}}\left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}}(\bar{\alpha})^{\frac{\rho}{\rho-1}}\right]^{-\frac{1+\rho}{\rho}}. \tag{20}$$

See Appendix B for derivation details.

This completes the production function aggregation procedure. The derived distribution (20) characterizes firm-level heterogeneity in the effective AI quality level required to support a macro CES production function of the form (14), with economy-wide aggregate AI management and manual labor inputs and parameters θ_A, θ_M, ρ, obtained from micro-level Leontief production functions of the form (9).

The aggregation results in this section provide a foundation for linking micro-level technology deployment and organizational choices to macroeconomic production frameworks. By showing how heterogeneous firms with step-by-step Leontief technologies and endogenous task boundaries can be represented by a CES aggregator, our framework offers a micro-founded rationale for using aggregate CES production functions to study labor demand, substitution patterns, and productivity in economies adopting new AI technologies. This connection clarifies how firm-level decisions about automation, augmentation, and job design scale up to shape aggregate technological change.

7 Empirical Evaluation

In this section, we empirically test three predictions of our model: (1) AI-executed steps tend to appear next to each other in the production sequence, forming AI chains; (2) occupations in which AI-suitable steps are more dispersed have fewer AI-executed steps; and (3) the AI-execution status of a step's neighbors affects the likelihood that the step itself is executed by AI.

For our analysis, we construct a dataset that records three attributes of each step: its AI exposure status (exposed or unexposed to AI), its realized mode of execution (manual, AI-augmented, or AI-automated), and its position in the production sequence. To create this dataset, we combine data from four sources:

1. The O*NET 27.3 Database (National Center for O*NET Development, 2023),
2. Human-generated labels for AI exposure of O*NET tasks from Eloundou et al. (2024),
3. Realized AI execution mode of O*NET tasks from the Anthropic Economic Index (Handa et al., 2025),
4. A GPT-generated task ordering for each O*NET occupation.

The structure of our theoretical framework closely aligns with the structure of the O*NET dataset. Each production step in the model corresponds to an O*NET task, and each job corresponds to an O*NET occupation. In the absence of AI, a step (that is, an O*NET task) is executed manually and maps directly into a task under the model's definition. Thus, without AI, an O*NET task can be viewed as both a step and a task, which matches the standard treatment of tasks in the O*NET dataset. Later when we discuss AI execution, however, multiple steps, corresponding to a subset of O*NET tasks, may be chained together and executed jointly by AI, forming a composite task per the model's definition. Throughout this section, we use the terms step, O*NET task, and task interchangeably and expect the reader to keep these distinctions in mind. We similarly use the terms job and occupation interchangeably without restating the mapping each time.

We use the May 2023 release of O*NET. This version contains roughly 18,000 tasks assigned to more than 850 U.S. occupations. For AI exposure measures, we use human-generated labels from Eloundou et al. (2024). Unless explicitly mentioned otherwise, in all analyses we treat tasks with a human-assigned E1 label as exposed to AI and those with E0 or E2 labels as unexposed to AI.[22] This gives a conservative measure of exposure to AI for all O*NET tasks.

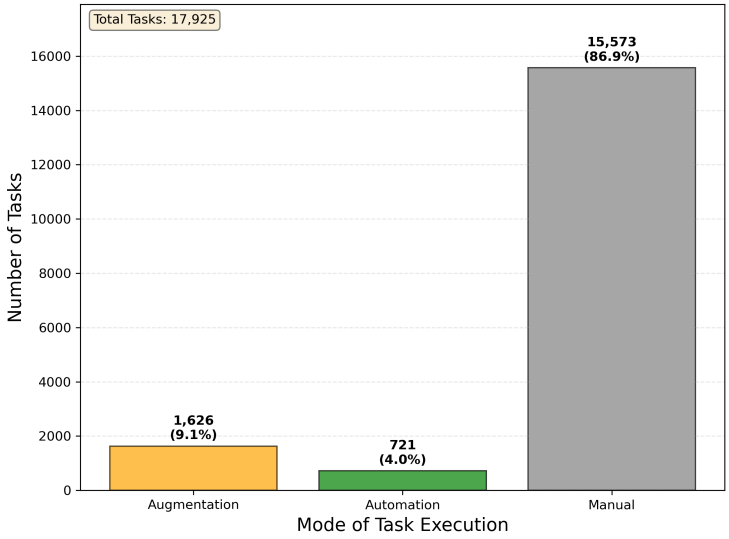
To obtain AI execution labels, we draw on Anthropic's Economic Index dataset (Handa et al., 2025), which classifies millions of Claude conversations into six categories: *"Validation", "Task Iteration", "Learning", "Directive", "Feedback Loop"*, and *"Filtered."* Conversations in the first three categories (Validation, Task Iteration, Learning) are treated by Anthropic as AI-augmenting activities, whereas those in Directive and Feedback Loop are recognized as AI-automating activities. The sixth category, Filtered, corresponds to conversations that either could not be classified or whose actual category could not be disclosed for privacy reasons.

Anthropic reports the share of matched conversations falling into each of the six categories for a subset of O*NET tasks. In total, conversations are linked to 3,364 tasks, of which 1,017 have 100% of their conversations filtered, leaving 2,347 tasks with at least one non-filtered conversation. Following Anthropic's definitions of AI-augmenting and AI-automating activities, we assign each of these 2,347 tasks an AI execution label based on the majority share of its non-filtered conversations across the augmenting versus automating categories. Appendix Table C.2 provides several example tasks labeled using this procedure.

Finally, we treat tasks that do not appear in the Anthropic dataset, as well as those with 100% filtered conversations, as manual tasks. Figure 6 shows the distribution of task execution modes in our dataset. Of the 872 occupations, 605 (69%) contain at least one AI-exposed task and 555 (64%) contain at least one AI-executed task. Figure 7 shows the distribution of occupations by the fraction of their tasks that are exposed to AI (left) and executed by AI (right).

[22] Eloundou et al. (2024) define E1 tasks as those that an AI can perform in at least half the time required by a human while preserving human-level output quality. E2 tasks meet the same time-saving and quality thresholds but require additional software or tools to fully leverage AI capabilities. Any task that is neither E1 nor E2 is labeled E0.

Figure 6: Distribution of Modes of Task Execution in the Dataset

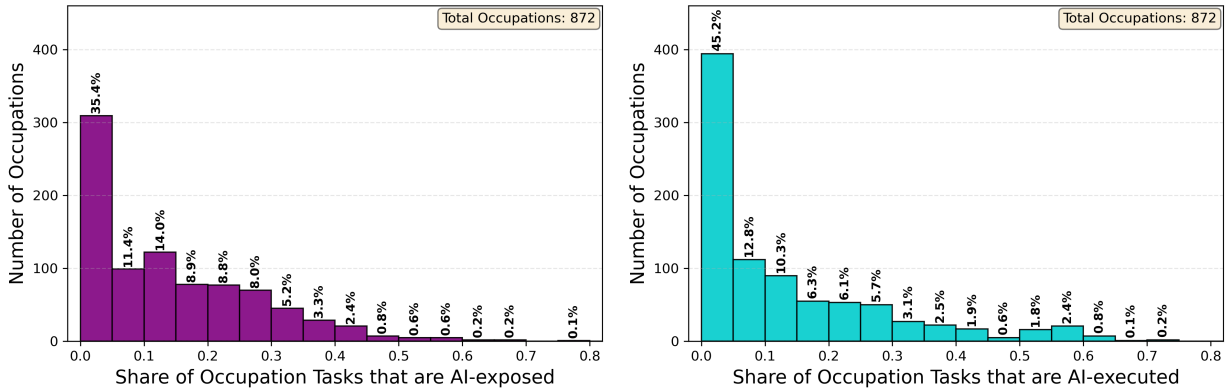


Notes: The automation and augmentation labels are drawn from Anthropic's Economic Index dataset, and the universe of tasks comes from the May 2023 O*NET release.

Figure 7: Distribution of Share of Occupation Tasks Exposed to and Executed by AI

(a) AI Exposure

(b) AI Execution



Notes: Percentages above bars denote fraction of observations corresponding to that bar out of all 872 occupations.

As the final input into our analyses, we generate a workflow sequence for every O*NET occupation. Specifically, we prompt GPT-5-mini through Expected Parrot (Horton and Horton, 2024) with the complete set of tasks within each occupation and ask it to determine the most reasonable sequential ordering in which these tasks would be performed in practice.[23] Appendix Figures D.2,

[23]The prompt used to generate task sequences is given in Appendix E.

D.3, and D.4 show the resulting task sequences for three occupations: *Computer Programmers* with a high share of its tasks executed by AI, *Public Relations Specialists* with a moderate share, and *Electronic Equipment Installers and Repairers, Motor Vehicles* with no tasks executed by AI.

We validate these GPT-generated sequences in two ways. First, throughout our empirical analyses, we benchmark the outcomes of interest against those generated by placebo datasets obtained by randomly reshuffling task positions within each occupation. We find that patterns in the GPT-generated data differ sharply from those in the placebo datasets, indicating that GPT captures meaningful workflow structure rather than producing arbitrary task orderings. Second, in Data Appendix F, we show that our headline results are robust to the choice of prompt. Specifically, we evaluate ten deliberately different alternative prompt formulations and find substantial, though not perfect, overlap in pairwise task orderings across prompts.[24] We also find no evidence that prompt choice induces systematic differences in task orderings across the occupation groupings and measures used in our later analyses, suggesting that our results are unlikely to be driven by prompt-specific bias toward particular groups in the main prompt. Finally, we re-run all our headline analyses using task orderings from these alternative prompts and find patterns that closely mirror those under the main prompt.

A noteworthy feature of these task sequences is that the realized execution modes are broadly consistent with, but do not perfectly satisfy, our model's definition of AI chains. In particular, in the model an automated task can only be followed by another AI-executed task, either automated or augmented, whereas in the data we occasionally observe an automated task followed by a non-AI (i.e., manual) task. Task 5 in *Computer Programmers*' occupation (Appendix Figure D.2) and Task 3 in *Public Relations Specialists*' occupation (Appendix Figure D.3) are two such instances. This pattern is a consequence of natural definitional differences between our model and the Anthropic data. Specifically, the automated versus augmented distinction in our model is defined by production sequencing and task dependencies, whereas the Anthropic classifications are based on task-level exposure and feasibility and are agnostic to workflow position. As a result, tasks labeled as automated in the data may still be followed by manual steps, leading to apparent deviations from the model's sequencing restriction without contradicting its underlying logic. Therefore, when considering the structure of AI chains in the remainder of this section, we do not distinguish between tasks labeled as automated versus augmented in the empirical data; rather, we treat both labels as indicating AI execution.

Before discussing the findings, it is worth emphasizing two points about the relevance of our results. First, note that we use a conservative measure of AI involvement in task execution. Recall that we label tasks with fully filtered conversations as manual, even though we know that all of them were executed in some way using Claude. These tasks account for roughly one third of all Anthropic-mapped tasks, and excluding them limits our statistical power. Nevertheless, the remaining two thirds prove sufficient for revealing the patterns predicted by our framework, suggesting that what we report should be viewed as a conservative signal of potentially much stronger effects.

A second concern is that workers may use different AI tools for different purposes, for example Claude for writing and ChatGPT for coding, which could potentially undermine our results. While we cannot directly rule out such heterogeneous tool use given that we only observe Claude usage, our analyses are not tied to particular industries, occupations, or tasks, which makes them less vulnerable to missing isolated pockets of tool-specific usage.[25] Nevertheless, wherever applicable,

[24] Across all occupations and task pairs, the average Kendall's τ is 0.6.

[25] For our results to be entirely spurious, one would have to believe not only that no one used Claude for a large subset

we verify that our results are not driven by any narrow subset of occupations or tasks. The patterns we document appear consistently across broad occupation groups, even if somewhat stronger in some groups than others, reflecting heterogeneous impacts of AI. Finally, while we do not claim that these relationships are causal, taken together they provide strong suggestive evidence that the mechanisms highlighted by our model are relevant in practice and that our findings reflect broad patterns of AI execution rather than Claude-specific usage.

7.1 Prediction #1: Tendency to Have Runs of Consecutive AI-executed Tasks

The first prediction of the model that we test is that, to leverage cost savings from forming longer AI chains, AI steps tend to appear in contiguous blocks rather than being scattered in the workflow. This prediction follows from our definition of AI chains (Definition 4), which requires every automated step to be followed by another automated or augmented step, and from the fragmentation argument, which suggests that for a given level of AI quality longer AI chains are more likely to emerge when AI-easy steps are clustered together in the workflow (see Examples 1 and 2 in Subsection 5.1).

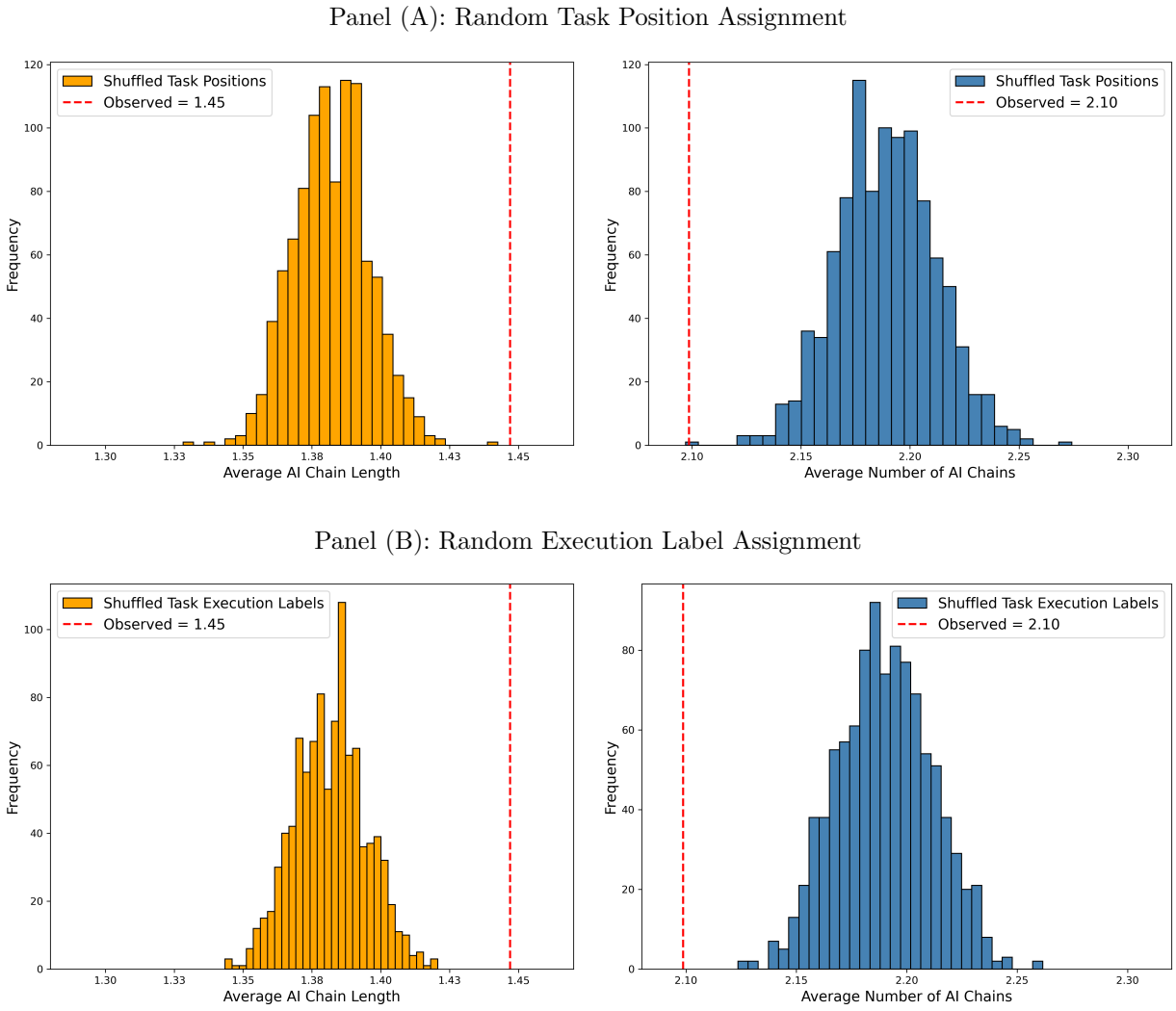
Instances of this pattern are evident in example task sequences shown in Appendix Figures D.2 and D.3. To test this systematically, we compute the average AI chain length and the average number of AI chains per occupation and compare them with two sets of placebo reshuffles of the original dataset. In the first placebo, we randomize the positions of tasks within each occupation's task sequence whereas in the second we randomize the assignment of AI execution labels across tasks in the entire dataset.

These placebo tests serve two purposes. First, and most importantly, because we do not have an external benchmark for these statistics, they verify that the patterns we observe in the actual data are meaningful rather than artifacts of randomness. Second, each placebo targets a distinct concern about our dataset. The randomized task position placebo tests whether the GPT-generated workflow sequences contain meaningful structure or behave like random permutations of tasks within occupations, while the execution label reassignment placebo tests whether Anthropic's AI execution labels are randomly scattered across tasks or instead exhibit meaningful co-occurrence within occupations.

For measurement, recall that we treat all automated and augmented tasks as a single type of AI task, with any consecutive run of AI-executed tasks constituting an AI chain. With this definition, the average AI chain length in the original dataset is 1.45, and the average number of AI chains per occupation is 2.10. Figure 8 plots the observed values in the actual dataset under the main prompt alongside histograms of the same two statistics calculated from 1,000 task position reshuffles (Panel A) and 1,000 random execution label assignments (Panel B) of the original dataset. Comparing the observed average AI chain length with the distributions generated by random task position and execution label assignments in the graphs on the left, we see that the average AI chain length in the original data is noticeably larger than in both placebo distributions. This indicates that the AI-executed tasks tend to cluster within occupations, forming longer AI chains than would arise under either type of random assignment. The results for the number of AI chains in the right-hand

of tasks during Anthropic's sample period but also that this missing class of tasks has distinctive characteristics, such as workflow position or AI-suitability, that differ from other tasks for which Claude is used across the economy. We do not have a reason to believe this is the case. If anything, specialization across tools would imply that Anthropic's data might understate the true set of AI-executed tasks in the economy, which would primarily reduce our statistical power rather than generate artificial patterns.

Figure 8: Average Length and Average Count of AI Chains: Actual versus Placebo Datasets



Notes: Red dashed lines show the observed average AI chain length (orange graphs on the left) and average AI chain count (blue graphs on the right) in the actual dataset under a more lenient definition of AI chains. Panel (A) shows the histogram of the two statistics for 1,000 placebo datasets in which the positions of tasks within each occupation are randomized. Panel (B) shows the histogram of the two statistics for 1,000 placebo datasets in which the execution labels of tasks in the entire dataset are randomized.

graphs tell a consistent story. Holding fixed the number of AI-executed tasks in the dataset, longer chains necessarily imply fewer separate chains. Accordingly, the right-hand graphs show that the observed number of AI chains per occupation is meaningfully smaller than what would be expected under random assignment in either case.

These findings also mitigate two potential concerns regarding the structure of our dataset. In both placebo tests, the average AI chain length and the number of AI chains observed in the actual data lie at the extreme tails of the placebo distributions. First, this implies that the

GPT-generated workflow sequences are unlikely to be chosen arbitrarily and without logic. If the generated sequences were random, the average AI chain length and the number of AI chains would lie closer to the center of the distributions generated by arbitrary within-occupation reordering of tasks. Second, the results indicate that the clustering of AI-executed tasks in the Anthropic data is not driven by chance co-occurrence of execution labels, but instead reflects systematic patterns within occupations across the economy. Together, these results show that the observed patterns of AI chaining capture meaningful workflow structure rather than artifacts of random task ordering or random AI execution label assignment.

7.2 Prediction #2: Dispersion of AI-able Tasks Affects AI Execution Outcomes

The second prediction of the model that we test is that the extent of realized AI execution in an occupation is determined not just by the AI exposure status of its steps, but also by how clustered or dispersed those AI-able steps are in the workflow. To examine this, we ask whether occupations with similar shares of their tasks exposed to AI exhibit different shares of realized AI-executed tasks depending on how dispersed their AI-able tasks are in the production sequence. Notably, this level of dispersion is related to the fragmentation index from Section 5.1; we will make use of a slightly modified metric (which we call the *empirical fragmentation index*, defined in more detail below) that can be calculated from our dataset. Specifically, we estimate the following regression:

$$\text{ai\_execution}_o = \beta_0 + \beta_1 \, \text{ai\_exposure}_o + \beta_2 \, \text{empirical\_fragmentation\_index}_o + \varepsilon_o, \tag{21}$$

where $ai\_execution_o$ is the share of tasks executed by AI in occupation o, $ai\_exposure_o$ is the share of AI-exposed tasks, and $empirical\_fragmentation\_index_o$ measures how dispersed the AI-able tasks are across the occupation's workflow.

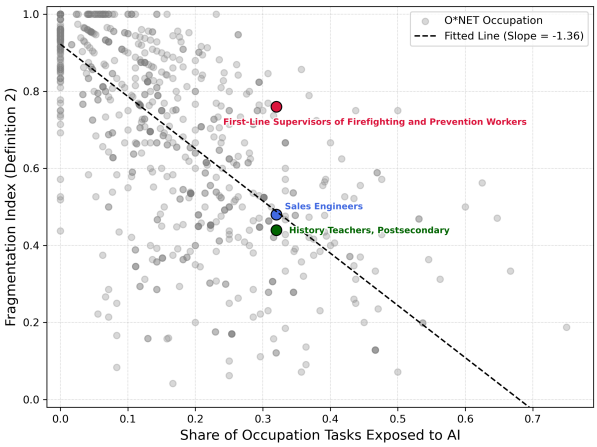
The model predicts that occupations with higher AI exposure should have a higher share of steps executed by AI ($\beta_1 > 0$). Conditional on exposure, it also predicts that when AI-able steps are more dispersed in the workflow, the share of realized AI-executed steps should be lower ($\beta_2 < 0$). The intuition is that, for a given level of AI quality (parameter α in the model), AI chains are more likely to form when AI-easy steps appear next to each other, as discussed in Examples 1 and 2 of Subsection 5.1. When such clustering occurs, nearby steps that would otherwise remain manual may instead be automated as part of a chain.

To construct an empirical measure of fragmentation, we abstract from heterogeneity in tasks' underlying cost profiles, which are unobserved in our data, and focus solely on their positioning within the workflow. We define an occupation's empirical fragmentation index (EFI) as follows. Assuming all AI-exposed steps turn into AI-executed tasks, empirical fragmentation index is the ratio of the number of *potential* tasks in the sense of our theoretical framework to the total number of steps, which is fixed and is measured by the raw count of O*NET tasks in that occupation. The following example clarifies the definition. Consider an occupation with five O*NET tasks. If none of the five tasks are exposed to AI, then the number of tasks in the theoretical sense is five, and the EFI equals $5/5 = 1$. If instead two consecutive O*NET tasks are exposed to AI (and thus can form an AI chain when all AI-exposed tasks turn into AI-executed tasks), the number of tasks in the model falls to four and the EFI becomes $4/5 = 0.8$. Thus, the empirical fragmentation index decreases as more AI-exposed steps can potentially be consolidated into longer AI chains.

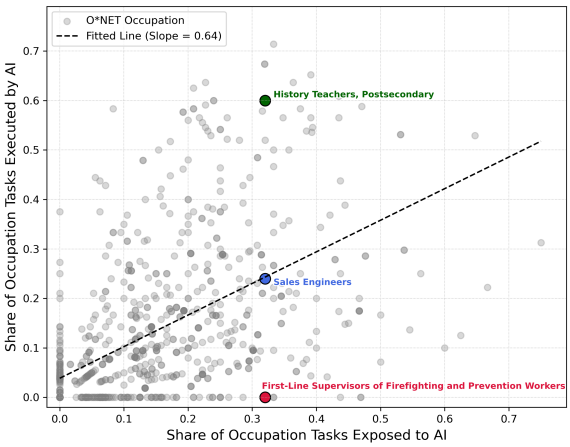
For the calculation of EFI, we treat all E1-exposed tasks as identical for the purpose of potential chaining and assume that all AI-exposed tasks could, in principle, be executed by AI. Because only

Figure 9: Relationships Between Occupational AI Exposure, Empirical Fragmentation, and AI Execution

(a) Empirical Fragmentation vs. AI Exposure



(b) AI Execution vs. AI Exposure



Notes: These graphs show how the fragmentation of AI-able tasks within an occupation's workflow shapes the conversion of AI-exposed tasks into AI-executed tasks. Each bin represents a single O*NET occupation in both panels. Panel (a) plots version 2 of the empirical fragmentation index against the share of AI-exposed tasks, whereas Panel (b) plots the share of realized AI-executed tasks against the share of AI-exposed tasks. All three highlighted occupations have 25 tasks in their production sequence and have 32% of their tasks exposed to AI.

14% of tasks in our dataset are E1-exposed, we also consider a broader definition of AI-able tasks when computing the empirical fragmentation. Specifically, we expand the set of AI-able tasks that can form AI chains to include both E1- and E2-exposed tasks, which together account for 44% of all tasks in the dataset. This expansion allows us to capture a wider range of tasks that may plausibly be executed by AI and yields a more informative measure of AI-able task fragmentation. Accordingly, we construct two versions of the empirical fragmentation index. Definition 1 is a stricter measure that permits only E1-exposed tasks to form AI chains. Definition 2 allows a broader set of AI-exposed tasks, including both E1- and E2-exposed tasks, to potentially form AI chains and therefore reflects a more lenient mapping from exposure to execution.

Figure 9 visualizes the relationship between occupational AI exposure, empirical fragmentation, and realized AI execution. Panel (a) plots the second definition of the empirical fragmentation index against the share of occupation tasks exposed to AI, while Panel (b) plots the realized share of tasks executed by AI against occupation-level AI exposure. The bins for three selected occupations are highlighted: *First-Line Supervisors of Firefighting and Prevention Workers, Sales Engineers,* and *History Teachers*. We focus on these three occupations for illustrative purposes, as they all contain the same number of tasks in their workflow, namely 25, and have 32% of their tasks exposed to AI.

Despite having identical exposure shares, these occupations differ in the dispersion of their AI-exposed tasks, as shown in Panel (a). In line with the model's prediction, these differences in dispersion translate into differences in realized AI execution outcomes, as shown in Panel (b). Specifically, the more clustered AI-exposed tasks are within an occupation (corresponding to a lower EFI), the larger is the share of its tasks executed by AI.

Table 3: Role of AI-able Task Fragmentation in Determining AI Execution Outcomes

	EFI Definition 1			EFI Definition 2		
	(1)	(2)	(3)	(4)	(5)	(6)
AI Exposure	0.66***	0.20**	0.18**	0.33***	0.14**	0.12**
	(0.08)	(0.08)	(0.08)	(0.06)	(0.06)	(0.06)
Empirical Fragmentation Index	0.04	-0.15	-0.01	-0.23***	-0.21***	-0.14***
	(0.20)	(0.16)	(0.16)	(0.03)	(0.04)	(0.04)
R-squared	0.31	0.63	0.73	0.37	0.66	0.74
Adj. R-squared	0.31	0.62	0.70	0.37	0.65	0.71
Observations	872	872	872	872	872	872
SOC Group Fixed Effect		Major	Minor		Major	Minor

Clustered standard errors in parentheses. $^{*}: p < 0.1$, $^{**}: p < 0.05$, $^{***}: p < 0.01$.

Notes: This table shows that occupation-level AI execution outcomes depend not only on the overall exposure of tasks to AI, but also on where AI-able steps are situated in the production sequence. The table reports results from estimating regression (21). The dependent variable in all specifications is the share of steps executed by AI in an occupation (ai_execution). The variable "AI Exposure" denotes the share of AI-exposed (E1) steps in the occupation, while the "Empirical Fragmentation Index" captures how dispersed AI-able steps are across the occupation's workflow. Definition 1 measures fragmentation based on potential AI chains formed exclusively by E1-exposed tasks, whereas Definition 2 measures fragmentation based on potential chains formed by both E1- and E2-exposed tasks.

To generalize the patterns illustrated in the figure, we estimate regression (21) using the two exposure-based empirical fragmentation measures. The results are reported in Table 3. For each definition, we estimate three specifications: without additional controls, with SOC major group fixed effects, and with SOC minor group fixed effects. We include SOC fixed effects to ensure that the estimated relationships are not driven by a narrow subset of occupation families. Across specifications, we find that a 10 percentage point increase in occupation-level AI exposure is associated with a 1.2 to 6.6 percentage point increase in the share of occupation tasks executed by AI. Consistent with the model's prediction, the coefficient on the empirical fragmentation index is negative in nearly all specifications, indicating that greater dispersion of AI-able steps is associated with lower realized AI execution at the job level after controlling for AI exposure. The only exception arises when we use the stricter fragmentation measure in Definition 1 without SOC controls in column (1). Under this conservative measurement of the hypothetical exposure-to-execution mapping, the fragmentation estimates are imprecise and statistically insignificant because the limited number of E1-exposed tasks sharply restricts the set of steps that can potentially form an AI chain. When we instead use the more permissive measure in Definition 2, which allows a broader set of AI-exposed tasks to form potential chains when computing fragmentation, the estimated coefficients on EFI are negative and statistically significant at the 1% level across all specifications.

To ensure that our results are not driven by how fragmentation is measured using the AI exposure classifications of Eloundou et al. (2024), we repeat the analysis as a robustness check

using two additional measures of task fragmentation constructed directly from realized AI execution outcomes, rather than from AI-exposed tasks. The corresponding regression results are reported in Appendix Table C.1, with complementary visual evidence provided in Appendix Figure D.5.

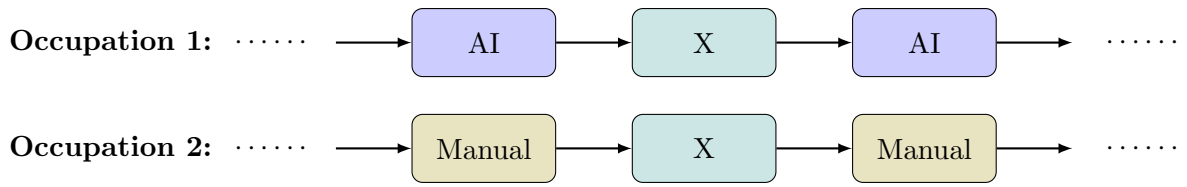
By construction, the execution-based EFI measures are mechanically related to the share of tasks executed by AI. Holding the task sequence fixed, an additional AI-executed task necessarily increases the occupation's AI execution share while weakly reducing its EFI.[26] As a result, these measures mechanically amplify the negative relationship between AI execution and fragmentation in the regression, as reflected in the estimated coefficients on the EFI reported in Appendix Table C.1. Accordingly, they are not intended to provide an independent characterization of the relationship between task fragmentation and AI execution.

Instead, these measures serve as a diagnostic of the *form* that AI adoption takes within occupations. The model predicts not only that greater clustering of AI-able tasks facilitates higher overall AI execution, but also that realized AI adoption should occur disproportionately through the extension of contiguous AI chains rather than through scattered, isolated task substitutions. If AI execution were independent across steps, then conditional on AI exposure, increases in execution would primarily appear as isolated events and would not substantially reduce fragmentation. The strong negative relationship observed between execution-based EFI and AI execution therefore indicates that AI adoption follows the clustered, chain-based pattern implied by the model. In this sense, the execution-based EFI measures function as a falsification-style check on the model's underlying mechanism rather than as a separate test of its predictions.

Taken together, the evidence from both exposure-based (ex-ante) and execution-based (ex-post) empirical fragmentation measures points to a common pattern. Occupations with greater AI exposure exhibit higher levels of realized AI execution, but the extent to which exposure translates into execution depends systematically on how AI-able steps are organized within the workflow. When AI-able steps are more clustered, realized AI execution is higher and occurs through contiguous chains rather than as isolated step substitutions.

7.3 Prediction #3: Adjacency to AI Tasks Increases Likelihood of AI Execution

The final prediction of the model that we test is that when the same step appears in two occupations, in one sitting between strong AI performers and in the other located between manual ones, it is more likely to be executed by AI in the first occupation. Intuitively, when a step is surrounded by AI-executed steps, there are local gains from grouping it with its neighbors as part of a long AI chain that are absent when the step is situated next to manual steps. For example, step X is more likely to be executed by AI in the first occupation below than in the second because, all else equal, the local gains from forming an AI chain containing all three steps are larger than those from performing only step X via AI in the second occupation:



[26]The reduction is weak because adding an isolated AI-executed task does not change fragmentation, whereas adding a task adjacent to an existing AI-executed task extends an AI chain and reduces fragmentation.

Testing this prediction requires identifying steps that appear across different occupations. Because O*NET tasks are occupation-specific, no single task is shared across occupations in the data. To work around this, we rely on O*NET's broader categories of Detailed Work Activities (DWAs), which group conceptually similar tasks across occupations. In total, the roughly 18,000 O*NET tasks are mapped into 2,067 DWAs.

In our main sample, we drop tasks that are mapped to multiple DWAs.[27] We further restrict attention to DWAs that contain tasks appearing in more than one occupation. Together, these restrictions reduce the sample to 10,708 tasks spread across 1,748 DWAs. We treat the remaining tasks as instances of the same step appearing across different occupations and estimate the following regression:

$$\Pr(\text{is\_ai}_t = 1 \mid X_t) = \Lambda\Big(\beta_0 + \beta_1 \,\text{prev2\_is\_ai}_t + \beta_2 \,\text{prev\_is\_ai}_t + \beta_3 \,\text{next\_is\_ai}_t + \beta_4 \,\text{next2\_is\_ai}_t + \varepsilon_t\Big), \tag{22}$$

where Λ is the logistic CDF and is_ai_t indicates whether step t is executed by AI. The variables $prev2\_is\_ai_t$ and $prev\_is\_ai_t$ equal 1 if the step two positions before or immediately before step t is AI-executed, respectively, and $next\_is\_ai_t$ and $next2\_is\_ai_t$ are defined analogously for the steps immediately after and two positions after step t. In the implementation we also control for the step t's AI exposure status and number of steps in the occupation it appears in.

Table 4 reports average marginal effects (AMEs) from estimating equation (22) under a series of increasingly restrictive specifications. Column (1) presents the baseline specification without additional controls. In this specification, having a task's immediate neighbors executed by AI is associated with a large and statistically significant increase in the probability that the task itself is AI-executed. More distant neighbors also exhibit positive effects, but their magnitudes are substantially smaller than those of the immediate neighbors.

Columns (2) and (3) add SOC major group and SOC minor group fixed effects, respectively. Once we control for occupation families, the estimated effect of immediate neighbors attenuates, and the coefficients on more distant neighbors become small and statistically insignificant. This pattern suggests that part of the baseline estimates reflect systematic differences across occupation families, while the effect of immediate neighbors remains present even within families.

In column (4), we include DWA fixed effects so that the variation comes from within detailed work activities, rather than being driven by systematic differences across DWAs that may exhibit distinct AI-ability characteristics. Under this specification, the effect of immediate neighbors attenuates relative to the baseline, and the influence of more distant neighbors effectively disappears. This indicates that a meaningful share of the baseline relationship operates at the level of the DWA, and that once comparisons are restricted to tasks within the same detailed work activity, the estimated effects are reduced but remain present for immediate neighbors.[28]

In a non-trivial number of cases, a DWA contains multiple tasks within the same occupation in the O*NET dataset. While our main sample retains the full set of such tasks, column (5) controls for the number of tasks associated with a task's DWA within the same occupation. This control is intended to mitigate concerns that tasks belonging to the same DWA and occupation may appear close together in the workflow and mechanically inflate the estimated impact of neighboring tasks on a task's likelihood of AI execution. The resulting estimates closely resemble those from the baseline, although the effects of more distant neighbors are weaker and less precisely estimated.

[27] A little over 20% of tasks are associated with more than one DWA in the O*NET dataset.

[28] We do not include simultaneous SOC occupation family and DWA fixed effects as doing so severely restricts the sample and we lack enough variation required for properly estimating the coefficients in those specifications.

Table 4: Effect of Neighboring Tasks' AI Execution Status on Task's AI Execution Likelihood

Specification	(1)	(2)	(3)	(4)	(5)	(6)
$(t-2)$ Task AI	0.07***	0.01	0.00	-0.01	0.06*	-0.01
	(0.02)	(0.01)	(0.01)	(0.03)	(0.03)	(0.03)
$(t-1)$ Task AI	0.12***	0.06***	0.05***	0.05**	0.13***	0.05**
	(0.02)	(0.01)	(0.01)	(0.02)	(0.03)	(0.02)
$(t+1)$ Task AI	0.12***	0.06***	0.05***	0.04**	0.10***	0.04**
	(0.02)	(0.01)	(0.01)	(0.02)	(0.02)	(0.02)
$(t+2)$ Task AI	0.05***	0.00	-0.01	0.00	0.04	0.00
	(0.02)	(0.01)	(0.01)	(0.02)	(0.03)	(0.02)
Pseudo R^2	0.112	0.173	0.171	0.196	0.030	0.197
Observations	10,708	10,708	9,861	4,096	4,096	4,096
SOC Group FE		Major	Minor			
DWA FE				Yes		Yes
NumTasks in DWA-Occupation Control					Yes	Yes

Clustered standard errors in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Notes: This table reports average marginal effects from estimating regression (22). The dependent variable in all specifications is an indicator for whether task t is AI-executed (is_ai_t). The sample is restricted to similar tasks across occupations, identified as tasks belonging to the same DWA in the O*NET dataset. All specifications control for the AI exposure status of task t and for the total number of tasks in task t's occupation. Standard errors are bootstrapped using $B = 200$ replications and clustered at the DWA level.

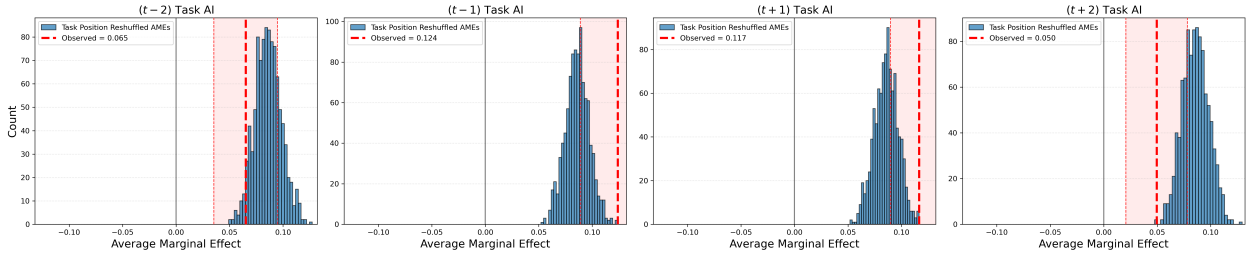
Finally, column (6) augments the last specification by including DWA fixed effects. Under this most restrictive specification, the estimated pattern mirrors that observed in columns (2) through (4), with small positive effects of immediate neighbors and negligible effects of more distant tasks.

Taken together, the estimates indicate that having a task's immediate neighbors executed by AI increases the probability that the task itself is AI-executed, whereas the mode of execution of more distant neighbors has little to no effect, especially after controlling for a range of relevant factors. This finding aligns with the relatively short AI chains observed in the data. Recall that the average AI chain length in our sample is only 1.45, implying that tasks two positions away rarely belong to the same chain as the focal task. Perhaps as AI quality improves longer chains may become more prevalent, in which case the execution status of more distant neighbors could begin to meaningfully influence a task's likelihood of AI execution as well.

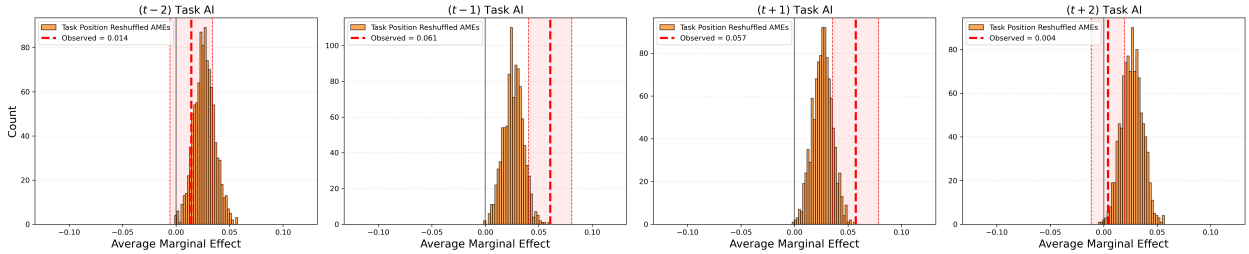
Because we do not have a benchmark for what constitutes a large versus small effect in our context, interpreting the magnitudes in a vacuum is difficult. To better understand these results and to check whether the observed patterns could have arisen by chance, we conduct a robustness exercise in which we re-estimate (22) for 1,000 placebo datasets constructed by randomizing the positions of tasks within occupations. The results are shown in Figure 10. In each graph, the red dashed line marks the AME estimated using the actual data (i.e., the values reported in Table 4) along with the 90% confidence interval around it. The blue bars in Panel (A) show the histogram of AMEs obtained from reshuffled datasets estimated using the baseline regression specification.

Figure 10: Effect of Neighboring Tasks' AI Execution Status on Task's AI Execution Likelihood

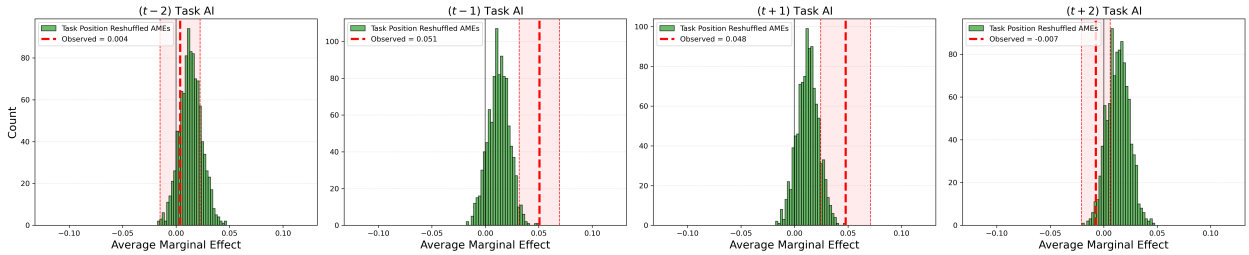
Panel (A): No Fixed Effects



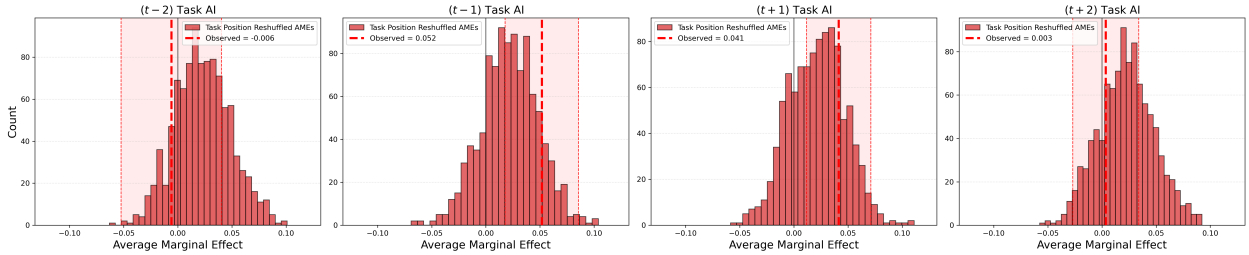
Panel (B): SOC Major Group Fixed Effects



Panel (C): SOC Minor Group Fixed Effects



Panel (D): Detailed Work Activity Fixed Effects



Notes: These graphs show that, among similar tasks appearing in multiple occupations, those whose immediate neighbors are AI-executed exhibit higher probabilities of being executed by AI, while more distant neighbors have little to no effect on a task's AI execution likelihood. The red dashed line in each graph indicates the observed average marginal effect of the corresponding variable in the original dataset reported in Table 4. The red shaded area marks the 90% confidence interval around the observed point estimates in the main sample. The histograms plot the distribution of average marginal effects across 1,000 randomized reshuffles of task positions within occupations. Panel (A) reports results from specification (22), while Panels (B), (C), and (D) augment the baseline specification with SOC major group, SOC minor group, and DWA fixed effects, respectively.

Panels (B), (C), and (D) report the corresponding histograms when SOC major group, SOC minor group, and DWA fixed effects are included, respectively.

Several patterns emerge from the graphs. First, across all four neighbors within each specification, the randomized distributions are centered at similar means and have similar shapes. This indicates that if ordering of tasks in the workflow had no effect on AI execution outcomes, each neighbor's mode of execution would contribute similarly, on average, to the AI execution probability of the task in question.

Second, in all panels, the observed marginal effects for immediate neighbors (middle columns) in the actual data lie to the right of the randomized distributions, while the actual effects for farther neighbors (side columns) tend to lie to the left. This suggests that immediate neighbors exert a stronger positive influence on a task's AI execution likelihood than would be expected under random assignment, while farther neighbors exert less influence once the contribution of immediate neighbors is accounted for.

Third, consistent with the results in Table 4, once SOC occupation group and DWA fixed effects are included, the estimated effects of more distant neighbors collapse to essentially zero in Panels (B) through (D), whereas the effects of immediate neighbors remain positive and distinct from the placebo distributions. This is consistent with the interpretation that due to short length of AI chains farther neighbors do not contribute to the execution mode of the task in the middle.

Next, we conduct two robustness exercises. The first addresses a concern regarding the selection of similar tasks across occupations. Although DWAs are intended to group conceptually similar tasks, one might argue that even within a given DWA tasks may still somewhat differ in their objectives, execution nature, or required skills. To alleviate concerns about the set of DWA tasks not sharing similar profiles, we repeat the analysis using a more restrictive definition of task similarity within DWAs. Specifically, for each DWA, we ask GPT-5-mini to select the subset of tasks that are most similar in terms of skill requirements and execution complexity.[29] This procedure yields, for each DWA, a set of highly comparable tasks that appear across different occupational contexts.[30] In this robustness test we treat these tasks as instances of the same underlying step appearing across occupations and re-estimate regression (22) on this restricted sample. The resulting estimates, reported in Appendix Table C.6 and Appendix Figure D.6, closely mirror the baseline patterns given above in both sign and magnitude.

The second robustness exercise examines a stricter notion of AI execution that is directly implied by our model, under which automated steps can only precede other AI-executed steps, whether automated or augmented. Accordingly, we repeat the analysis focusing specifically on AI automation rather than AI execution more broadly. In this scenario, the dependent variable in equation (22) is an indicator for whether step t is AI-automated (is_automated_t) instead of AI-executed (is_ai_t). The corresponding estimates for the main sample are reported in Appendix Table C.7 and Appendix Figure D.7, while results for the GPT-filtered sample are reported in Appendix Table C.8 and Appendix Figure D.8.[31] Under this stricter specification, the estimated effects are smaller in magnitude and often statistically insignificant, reflecting the more limited prevalence of AI automation in the data. Nevertheless, the signs and qualitative patterns of the estimates remain consistent

[29] The prompt used to identify similar DWA tasks is provided in Appendix E.

[30] Panel (A) of Appendix Table C.3 presents example DWAs and their associated O*NET tasks across occupations, along with execution modes. Panel (B) reports the subset of tasks retained after applying the similarity filter.

[31] Because relatively few tasks in the data are AI-automated, and because we further restrict attention to highly similar tasks across occupations, we retain tasks associated with multiple DWAs in the original O*NET dataset and randomly assign each such task to one of its associated DWAs for these exercises.

with those obtained for AI execution.

8 Conclusion

This paper develops a task-based framework for understanding how advances in AI reshape automation, the division of labor, firms' organizational structure, and production more generally. By modeling production as a sequence of steps that can be aggregated into tasks and then into jobs, we show how firms optimally allocate work between humans and AI (whether in augmented or fully automated form) when doing so requires trading off gains from specialization against coordination frictions. A central feature of the model is the possibility of AI chains, in which multiple steps are executed by AI and only the final output is verified by a human. Chaining can overturn step-level comparative advantage logic in factor assignment and non-linearly amplify the effects of marginal improvements in AI quality.

We characterize short-run AI deployment decisions when job boundaries, and therefore the set of tasks assigned to each worker, are fixed, and the long-run joint optimization of job design and AI deployment strategy. We show that improvements in AI quality can induce discrete reorganizations of work. A key implication is that marginal increases in AI quality may generate little or no cost savings until a threshold is crossed, after which the optimal production structure changes discontinuously. This helps explain why firms invest so heavily in AI capabilities even when short-run returns appear limited, and is consistent with the J-curve pattern of technology adoption (Brynjolfsson et al., 2021) in which early adoption raises costs before the anticipated gains from reorganization are realized.

We also show how the production functions derived from firm-level cost minimization can be mapped into a CES production function at the aggregate level when firms differ in how they deploy a common, general-purpose AI technology. Our framework therefore also allows studying the aggregate productivity and labor demand implications of improvements in AI quality through a micro-foundation for how individual firms respond to such advances.

We complement the theoretical analysis with empirical evidence consistent with three core mechanisms implied by the model. Using a task-level dataset of AI exposure and AI execution outcomes, we find that AI-executed steps tend to appear in contiguous AI chains, that occupations with more fragmented AI-exposed steps in their workflow exhibit lower realized AI execution conditional on exposure levels, and that when comparable steps appear across occupations those adjacent to AI-executed neighbors are significantly more likely to be executed by AI themselves. These patterns align with the central economic forces highlighted by the model: the importance of positional relationships among steps, the role of AI-able step dispersion in shaping execution outcomes, and the local complementarities created by AI chains.

References

Acemoglu, Daron, "The Simple Macroeconomics of AI," *Economic Policy*, 2025, *40* (121), 13–58.

_ **and David Autor**, "Skills, Tasks and Technologies: Implications for Employment and Earnings," in "Handbook of Labor Economics," Vol. 4, Elsevier, 2011, pp. 1043–1171.

_ **and Pascual Restrepo**, "The Race between Man and Machine: Implications of Technology for Growth, Factor Shares, and Employment," *American Economic Review*, June 2018, *108* (6), 1488–1542.

_ **and** _ , "Automation and New Tasks: How Technology Displaces and Reinstates Labor," *Journal of Economic Perspectives*, May 2019, *33* (2), 3–30.

_ **and** _ , "Tasks, Automation, and the Rise in U.S. Wage Inequality," *Econometrica*, 2022, *90* (5), 1973–2016.

_ **, David Autor, Jonathon Hazell, and Pascual Restrepo**, "Artificial Intelligence and Jobs: Evidence from Online Vacancies," *Journal of Labor Economics*, 2022, *40* (S1), S293–S340.

_ **, Fredric Kong, and Pascual Restrepo**, "Tasks At Work: Comparative Advantage, Technology and Labor Demand," Working Paper 32872, National Bureau of Economic Research August 2024.

Agrawal, Ajay, Joshua S Gans, and Avi Goldfarb, "Exploring the Impact of Artificial Intelligence: Prediction versus Judgment," *Information Economics and Policy*, 2019, *47*, 1–6.

Autor, David and Neil Thompson, "Expertise," *Journal of the European Economic Association*, 06 2025, *23* (4), 1203–1271.

Autor, David H, Frank Levy, and Richard J Murnane, "The Skill Content of Recent Technological Change: An Empirical Exploration," *The Quarterly Journal of Economics*, 2003, *118* (4), 1279–1333.

Baqaee, David Rezza and Emmanuel Farhi, "JEEA-FBBVA Lecture 2018: The Microeconomic Foundations of Aggregate Production Functions," *Journal of the European Economic Association*, 10 2019, *17* (5), 1337–1392.

Becker, Gary S and Kevin M Murphy, "The Division of Labor, Coordination Costs, and Knowledge," *The Quarterly Journal of Economics*, 1992, *107* (4), 1137–1160.

Bloom, Nicholas, Raffaella Sadun, and John Van Reenen, "Management as a Technology?," *The Quarterly Journal of Economics*, 2016, *132* (3), 1–60.

Bonney, Kathryn, Cory Breaux, Cathy Buffington, Emin Dinlersoz, Lucia S Foster, Nathan Goldschlag, John C Haltiwanger, Zachary Kroff, and Keith Savage, "Tracking Firm Use of AI in Real Time: A Snapshot from the Business Trends and Outlook Survey," Working Paper 32319, National Bureau of Economic Research April 2024.

Bresnahan, Timothy F, Erik Brynjolfsson, and Lorin M Hitt, "Information Technology, Workplace Organization, and the Demand for Skilled Labor: Firm-level Evidence," *The Quarterly Journal of Economics*, 2002, *117* (1), 339–376.

Brynjolfsson, Erik, Daniel Rock, and Chad Syverson, "The Productivity J-Curve: How Intangibles Complement General Purpose Technologies," *American Economic Journal: Macroeconomics*, January 2021, *13* (1), 333–72.

_ , **Tom Mitchell, and Daniel Rock**, "What Can Machines Learn, and What Does It Mean for Occupations and the Economy?," in "AEA Papers and Proceedings," Vol. 108 American Economic Association 2014 Broadway, Suite 305, Nashville, TN 37203 2018, pp. 43–47.

Coase, Ronald H, "The Nature of the Firm," *Economica*, 1937, *4* (16), 386–405.

Dell'Acqua, Fabrizio, Edward McFowland III, Ethan R Mollick, Hila Lifshitz-Assaf, Katherine Kellogg, Saran Rajendran, Lisa Krayer, François Candelon, and Karim R Lakhani, "Navigating the Jagged Technological Frontier: Field Experimental Evidence of the Effects of AI on Knowledge Worker Productivity and Quality," *Harvard Business School Technology & Operations Mgt. Unit Working Paper*, 2023, (24-013).

Deming, David J, "The Growing Importance of Social Skills in the Labor Market," *The quarterly journal of economics*, 2017, *132* (4), 1593–1640.

Dessein, Wouter and Tano Santos, "Adaptive Organizations," *Journal of Political Economy*, 2006, *114* (5), 956–995.

Eloundou, Tyna, Sam Manning, Pamela Mishkin, and Daniel Rock, "GPTs are GPTs: Labor Market Impact Potential of LLMs," *Science*, 2024, *384* (6702), 1306–1308.

Felipe, Jesus and Franklin M Fisher, "Aggregation in Production Functions: What Applied Economists Should Know," *Metroeconomica*, 2003, *54* (2-3), 208–262.

Felten, Edward, Manav Raj, and Robert Seamans, "Occupational, Industry, and Geographic Exposure to Artificial Intelligence: A Novel Dataset and its Potential Uses," *Strategic Management Journal*, 2021, *42* (12), 2195–2217.

Fisher, Franklin M, "Embodied Technical Change and the Existence of an Aggregate Capital Stock," *The Review of Economic Studies*, 1965, *32* (4), 263–288.

Frey, Carl Benedikt and Michael A Osborne, "The Future of Employment: How Susceptible are Jobs to Computerisation?," *Technological Forecasting and Social Change*, 2017, *114*, 254–280.

Furman, Jason and Robert Seamans, "AI and the Economy," *Innovation Policy and the Economy*, 2019, *19*, 161–191.

Gans, Joshua S and Avi Goldfarb, "O-Ring Automation," Working Paper 34639, National Bureau of Economic Research January 2026.

Garicano, Luis, "Hierarchies and the Organization of Knowledge in Production," *Journal of Political Economy*, 2000, *108* (5), 874–904.

_ **and Esteban Rossi-Hansberg**, "The Organization of Work in the Knowledge Economy," *Econometrica*, 2006, *74* (4), 1199–1227.

Handa, Kunal, Alex Tamkin, Miles McCain, Saffron Huang, Esin Durmus, Sarah Heck, Jared Mueller, Jerry Hong, Stuart Ritchie, Tim Belonax, Kevin K. Troy, Dario Amodei, Jared Kaplan, Jack Clark, and Deep Ganguli, "Which Economic Tasks are Performed with AI? Evidence from Millions of Claude Conversations," 2025.

Horton, John and Robin Horton, "EDSL: Expected Parrot Domain Specific Language for AI Powered Social Science," Whitepaper, Expected Parrot 2024.

Houthakker, Hendrik S, "The Pareto Distribution and The Cobb-Douglas Production Function in Activity Analysis," *The Review of Economic Studies*, 1955, *23* (1), 27–31.

Ide, Enrique and Eduard Talamàs, "Artificial Intelligence in the Knowledge Economy," *Journal of Political Economy*, 2025, *133* (12), 3762–3800.

Kremer, Michael, "The O-ring Theory of Economic Development," *The Quarterly Journal of Economics*, 1993, *108* (3), 551–575.

Leontief, Wassily, "Introduction to a Theory of the Internal Structure of Functional Relationships," *Econometrica, Journal of the Econometric Society*, 1947, pp. 361–373.

Levhari, David, "A Note on Houthakker's Aggregate Production Function in a Multifirm Industry," *Econometrica*, 1968, pp. 151–154.

McElheran, Kristina, J. Frank Li, Erik Brynjolfsson, Zachary Kroff, Emin Dinlersoz, Lucia Foster, and Nikolas Zolas, "AI Adoption in America: Who, What, and Where," *Journal of Economics & Management Strategy*, 2024, *33* (2), 375–415.

_ , Mu-Jeung Yang, Zachary Kroff, and Erik Brynjolfsson, "The Rise of Industrial AI in America: Microfoundations of the Productivity J-curve (s)," Technical Report 2025.

Milgrom, Paul and John Roberts, "The Economics of Modern Manufacturing: Technology, Strategy, and Organization," *The American Economic Review*, 1990, pp. 511–528.

_ and _ , "Complementarities and Fit Strategy, Structure, and Organizational Change in Manufacturing," *Journal of Accounting and Economics*, 1995, *19* (2-3), 179–208.

Murphy, Kevin Miles, "Specialization and Human Capital." PhD dissertation, The University of Chicago 1986.

National Center for O*NET Development, "O*NET 27.3 Database," `https://www.onetcenter.org/database.html` 2023. U.S. Department of Labor, Employment and Training Administration.

Sato, Kazuo, "Micro and Macro Constant-Elasticity-of-Substitution Production Functions in a Multifirm Industry," *Journal of Economic Theory*, 1969, *1* (4), 438–453.

_ , *Production Functions and Aggregation*, North-Holland Amsterdam, 1975.

Tambe, Prasanna and Lorin M. Hitt, "The Productivity of Information Technology Investments: New Evidence from IT Labor Data," *Information Systems Research*, 2012, *23* (3), 599–617.

Tomlinson, Kiran, Sonia Jaffe, Will Wang, Scott Counts, and Siddharth Suri, "Working with AI: Measuring the Applicability of Generative AI to Occupations," *arXiv preprint arXiv:2507.07935*, 2025.

Trammell, Philip, "Workflows and Automation," Working Paper, Digital Economy Lab, Stanford University 2026.

Webb, Michael, "The Impact of Artificial Intelligence on the Labor Market," *Available at SSRN 3482150*, 2020.

Williamson, Oliver E, "The Vertical Integration of Production: Market Failure Considerations," *The American Economic Review*, 1971, *61* (2), 112–123.

_ , "Transaction-Cost Economics: The Governance of Contractual Relations," *The Journal of Law and Economics*, 1979, *22* (2), 233–261.

A Technical Details – Fragmentation Index

In this appendix we prove Proposition 5, which relates the cost of the optimal short-term AI deployment strategy for a fixed job to the fragmentation index of that job. Intuitively, we expect a production process in which automatable tasks are clustered together will benefit more from AI deployment than one where steps with high and low levels of automation susceptibility are interspersed. This is due to the potential benefits from AI chains.

First recall the definition of fragmentation index for a given step sequence $\mathcal{S} = (s_1, \ldots, s_m)$. As a reminder, we assume for Proposition 5 that $t_i^A = 1$ for all s_i. Consider a random process in which each step s_i succeeds independently with probability q_i; any step that does not succeed is said to fail. Write F for the set of steps that fail, and $\mathcal{C} = \{C_1, \ldots, C_k\}$ for the random variable representing the collection of maximal connected components of non-failed steps. The weight of each $C_j \in \mathcal{C}$ is defined to be $\omega(C_j) = \min\{1, \sum_{s_i \in C_j} t_i^M\}$. That is, each C_j has weight 1 unless the sum of the manual time costs for each of its steps is less than 1 (due to our assumption that AI management costs are normalized to 1).

We now introduce some notation: for each s_i, write $t_i^* = \min\left\{t_i^M, \frac{t_i^A}{q_i}\right\}$. That is, t_i^* is the minimum cost to implement step s_i individually (whether manually or through AI augmentation). Given a realization of $\mathcal{C}$ and F, we define the realized fragmentation to be

$$\sum_{s_i \in F} t_i^* + \sum_{C_j \in \mathcal{C}} \omega(C_j).$$

The *fragmentation index* is defined to be the expected value of the realized fragmentation.

We now fix the sequence $\mathcal{S} = (s_1, \ldots, s_m)$, write FI for the fragmentation index of $\mathcal{S}$, and let OPT be the cost of the optimal (i.e., time-minimizing) task structure for those steps.

We begin by showing that FI is not much larger than OPT.

Proposition 6. $FI \leq \frac{5}{4} OPT$.

Proof. Fix an arbitrary task sequence $\mathcal{T} = (T_1, \ldots, T_n)$. Partition this into two sets of tasks: $\mathcal{T}_M$ containing all (singleton) tasks that are completed manually, and $\mathcal{T}_A$ containing all other tasks (consisting of AI-augmented singletons and AI chains). We first claim that

$$FI \leq \sum_{T_j \in \mathcal{T}_A} \left(1 + \sum_{s_i \in T_j} (1 - \alpha^{d_i})(1 + t_i^*)\right) + \sum_{(s_i) \in \mathcal{T}_M} t_i^M.$$

To see why, first note that for any human-executed task $(s_i) \in \mathcal{T}_M$, either the task fails in the realization of F (in which case it contributes $t_i^* \leq t_i^M$ to FI), or it succeeds (in which case it appears in some connected component C_j, contributing at most t_i^M to the component's weight $\omega(C_j)$).

Next consider the AI-assisted tasks. For any $T_j \in \mathcal{T}_A$, consider the set of steps in T_j that fail in any given realization of F. If no steps in T_j fail, then T_j collectively contributes at most 1 to the realized fragmentation (since all $s_i \in T_j$ lie in the same connected component of non-failed tasks). Otherwise, each failed task $s_i \in T_j$ contributes t_i^* to the realized fragmentation (for itself, recalling that t_i^* is the minimal cost of executing s_i on its own) plus an additional value of at most 1 for the weight of a potential new connected component of non-failed tasks. As each task $s_i \in T_j$ fails independently with probability $(1 - \alpha^{d_i})$, we get that the contribution of tasks in T_j to the fragmentation index is at most $1 + \sum_{s_i \in T_j} (1 - \alpha^{d_i})(1 + t_i^*)$ as claimed.

On the other hand, recall that if we take $\mathcal{T}$ to be the cost-minimizing task sequence, then

$$OPT = \sum_{T_j \in \mathcal{T}_A} \alpha^{-d(T_j)} + \sum_{(s_i) \in \mathcal{T}_M} t_i^M$$

by definition, where we write $d(T_j) = \sum_{s_i \in T_j} d_i$ for the total difficulty of steps in task T_j.

Comparing our expression for OPT with our bound on FI, we see that each task in $\mathcal{T}_M$ contributes the same amount to each, whereas each $T_j \in \mathcal{T}_A$ contributes $1+\sum_{s_i \in T_j}(1-\alpha^{d_i})(1+t_i^*)$ to the former and $\alpha^{-d(T_j)}$ to the latter. We will show that, for each $T_j \in \mathcal{T}_A$, $1+\sum_{s_i \in T_j}(1-\alpha^{d_i})(1+t_i^*) \leq \frac{5}{4}\alpha^{-d(T_j)}$, which will complete the proof.

First, since $t_i^* \leq \alpha^{-d_i}$ for each $s_i \in \mathcal{S}$, we have $1 + \sum_{s_i \in T_j}(1-\alpha^{d_i})(1+t_i^*) \leq 1 + \sum_{s_i \in T_j}(1-\alpha^{d_i})(1+\alpha^{-d_i})$.

Next note that for any $x, y \in [0,1]$, $(1-xy)(1+\frac{1}{xy}) \geq (1-x)(1+\frac{1}{x}) + (1-y)(1+\frac{1}{y})$. (This can be checked by expanding and taking derivatives.) Repeatedly applying this fact, we get that $1+\sum_{s_i \in T_j}(1-\alpha^{d_i})(1+\alpha^{-d_i}) \leq 1+(1-\alpha^{d(T_j)})(1+\alpha^{-d(T_j)}) = 1+\alpha^{-d(T_j)}-\alpha^{d(T_j)}$ for any $T_j \in \mathcal{T}_A$. The ratio between this expression and $\alpha^{-d(T_j)}$ is maximized at $\alpha^{-d(T_j)} = 1/2$, achieving a value of $5/4$ as claimed. □

Note that this bound of $5/4$ is tight. Suppose we have 3 steps in $\mathcal{S}$: the first and last always succeed, and the second succeeds with probability $1/2$ and has a manual cost of 2. Then the optimal policy automates all three together for an expected cost of 2, whereas the fragmentation index is $(1/2) \times 1 + (1/2) \times (1+2+1) = 5/2$.

We next argue that F is not much smaller than OPT. We begin with an analysis under the assumption that $t_i^M \geq 1$ for all $s_i \in \mathcal{S}$. That is, for each step s_i, completing the step manually is at least as costly as managing an AI tool to complete the step in a single attempt (with guaranteed success).

Proposition 7. *Suppose $t_i^M \geq 1$ for all $s_i \in \mathcal{S}$. Then $FI \geq \frac{1}{4}OPT$.*

Proof. Consider the following task sequence. Beginning with the first step s_1, consider the maximal contiguous sequence of steps T whose success probability $\alpha^{d(T)}$ is greater than or equal to $1/2$. If that set is empty (i.e., the first step has success probability strictly less than $1/2$) then the first step is set to be completed individually, either manually or augmented, whichever is cheaper. In this case, we'll say the resulting singleton task is completed *individually*. Otherwise, the sequence $T = (s_1, \ldots, s_\ell)$ (which could still be a singleton task) is added to the task sequence as an AI chain; we call this a *non-individual* task. This process is then repeated beginning with the next step (which is s_2 in the former case, or $s_{\ell+1}$ in the latter case), until all steps have been added to a task. Call the resulting task sequence $\mathcal{T}$.

Let ALG denote the cost of the task sequence $\mathcal{T}$. Note that we must have $ALG \geq OPT$. We will argue that $FI \geq ALG/4$, which will prove the claim.

We first define some notation. Write $\mathcal{T}_I$ for the set of individual tasks in $\mathcal{T}$ and $\mathcal{T}_{NI}$ for the set of non-individual tasks. Note that $\alpha^{d_i} < 1/2$ for all $(s_i) \in \mathcal{T}_I$, by construction. Note also that $\mathcal{T}_M \subseteq \mathcal{T}_I$ (recalling that $\mathcal{T}_M$ is the set of all singleton tasks executed manually), and this containment may be strict if $\mathcal{T}_I$ contains singleton tasks that are augmented. We emphasize that

$\mathcal{T}_{NI}$ may still include singleton tasks, but any singleton task $(s_i) \in \mathcal{T}_{NI}$ must have the property that $\alpha^{d_i} \geq 1/2$.

If the final task s_m from $\mathcal{S}$ is contained in some $T_j \in \mathcal{T}_{NI}$, we say that T_j is the *terminal* task. All other tasks in $\mathcal{T}_{NI}$ are non-terminal tasks. Note that if the final step is a singleton task in $\mathcal{T}_I$ then no task is terminal. For each non-terminal task $T_j \in \mathcal{T}_{NI}$, we'll write $\overline{T}_j$ to denote T_j plus the first step that follows T_j. For example, if $T_j = (s_i, \ldots, s_\ell)$, then $\overline{T}_j = (s_i, \ldots, s_\ell, s_{\ell+1})$. The set $\mathcal{T}_{NI}$ then has the property that for each task $T_j \in \mathcal{T}_{NI}$ we have $\alpha^{d(T_j)} \geq 1/2$, and for each non-terminal $T_j \in \mathcal{T}_{NI}$ we have $\alpha^{(d(\overline{T}_j))} < 1/2$.

We are now ready to relate FI and ALG. Let $k = |\mathcal{T}_{NI}|$ be the number of non-individual tasks. Note first that

$$ALG = \sum_{T_j \in \mathcal{T}_{NI}} \alpha^{-d(T_j)} + \sum_{(s_i) \in \mathcal{T}_I} t_i^* \leq 2k + \sum_{(s_i) \in \mathcal{T}_I} t_i^*.$$

This is because $\alpha^{d(T_j)} \geq 1/2$ for each $T_j \in \mathcal{T}_{NI}$, and hence $\alpha^{-d(T_j)} \leq 2$.

Next we bound FI. Note that the assumption $t_i^M \geq 1$ implies that, in any realization of the connected components of non-failed steps $\mathcal{C} = (C_1, \ldots, C_\ell)$, $\omega(C_j) = 1$ for all j. This implies that the realized fragmentation is equal to 1 plus, for each $s_i \in F$ (i.e., each step s_i that fails), a contribution of t_i^* plus an additional 1 in the event that $i > 1$ and the task immediately preceding s_i did not fail. (This additional cost of 1 accounts for creating a new connected component of non-failed steps ending with s_{i-1}.)

We claim that $FI \geq k/2 + \frac{1}{2}\sum_{(s_i) \in \mathcal{T}_I} t_i^*$. We will show this by charging the realized fragmentation to tasks in $\mathcal{T}_{NI}$ and $\mathcal{T}_I$, so that each $T_j \in \mathcal{T}_{NI}$ is charged at least 1 with probability at least $1/2$, and each $(s_i) \in \mathcal{T}_I$ is charged at least t_i^* with probability at least $1/2$.

To see this, let E_j denote the event that some step in $\overline{T}_j$ fails, for each non-terminal $T_j \in \mathcal{T}_{NI}$. Note that E_j occurs with probability at least $1/2$, since $\alpha^{d(\overline{T}_j)} < 1/2$. Also, if E_j occurs, then either a step $s_i \in T_j$ fails, or no step in T_j fails but the step immediately following T_j does. In the former case, we will charge the $t_i^* \geq 1$ contribution of s_i's failure to T_j. In the latter case, it must be that the step immediately preceding the failed task did not fail; so we will charge the additional contribution of 1 from s_i's failure (as described above) to T_j. Additionally, if there is a terminal T_j in $\mathcal{T}_{NI}$, then we charge the baseline 1 from the calculation of FI to that terminal T_j. Aggregating over all these cases, we have that at least 1 is charged to each set $T_j \in \mathcal{T}_{NI}$ with probability at least $1/2$. Finally, for each $(s_i) \in \mathcal{T}_I$ that fails (which occurs with probability at least $1/2$, by construction), we charge the t_i^* portion of its contribution to task (s_i). We conclude that $FI \geq k/2 + \frac{1}{2}\sum_{(s_i) \in \mathcal{T}_I} t_i^*$ as claimed.

But now since $FI \geq k/2 + \frac{1}{2}\sum_{(s_i) \in \mathcal{T}_I} t_i^*$ and $ALG \leq 2k + \sum_{(s_i) \in \mathcal{T}_I} t_i^*$, we conclude $FI \geq ALG/4$ as claimed. □

Can the approximation factor in Proposition 7 be improved? While we do not have a matching lower bound of 4, we do know that the approximation factor is at least $\frac{2}{3}(2+\sqrt{2}) \approx 2.276$. Here is an example. Suppose the problem instance is a sequence of m steps, each with success probability $1/\sqrt{2}$ and manual cost $\sqrt{2}$. The task sequence described in the proof of Proposition 7 then handles each task separately, for a total cost of $m\sqrt{2}$. The optimal task sequence must therefore perform at least this well. The fragmentation index is such that each task contributes $\sqrt{2}$ if it fails, plus 1 if the preceding task did not fail, for a total contribution of $(1-1/\sqrt{2})(\sqrt{2}+1(1/\sqrt{2}) = \frac{3}{2}(\sqrt{2}-1) \approx 0.6213$. As m grows large, the ratio between $m\sqrt{2}$ and $1 + m \times 0.6213$ approaches $\frac{2}{3}(2+\sqrt{2}) \approx 2.276$.

We can relax the assumption in Proposition 7 that $t_i^M \geq 1$ for all s_i, but at the cost of a worse approximation factor.

Proposition 8. *For general t_i^M (i.e., allowing $t_i^M < 1$ for some steps s_i), $FI \geq \frac{1}{8} OPT$.*

Proof. The argument follows the proof of Proposition 7, with the following changes.

First we make a slight change in the definition of the task sequence $\mathcal{T}$ that defines ALG. If a constructed task T_j has the property that $\sum_{s_i \in T_j} t_i^M < 1$ (which implies it is cheaper to run all steps of T_j manually than as an AI chain that is guaranteed to succeed), we switch to running the tasks of T_j manually in the task sequence. In our analysis, we still consider T_j to be a set in $\mathcal{T}_{NI}$; but its cost is taken to be $\sum_{s_i \in T_j} t_i^M$.

This modification changes our upper bound on the total cost of ALG to the following:

$$ALG \leq 2 \sum_{T_j \in \mathcal{T}_{NI}} \min \left\{ 1, \sum_{s_i \in T_j} t_i^M \right\} + \sum_{(s_i) \in \mathcal{T}_I} t_i^*.$$

Then, to bound FI, we claim that

$$FI \geq \frac{1}{4} \sum_{T_j \in \mathcal{T}_{NI}} \min \left\{ 1, \sum_{s_i \in T_j} t_i^M \right\} + \frac{1}{2} \sum_{(s_i) \in \mathcal{T}_I} t_i^*$$

which would prove the claim.

To see why this bound on FI holds, we employ a charging argument as before. Recall that for each non-terminal $T_j \in \mathcal{T}_{NI}$, the event E_j occurs with probability at least $1/2$. When it does, we charge to T_j the contribution t_i^* to FI from any $s_i \in F$ that lies in T_j. Furthermore, when E_j occurs, for each connected component C_ℓ that intersects T_j we additionally charge the contribution $\omega(C_\ell)$ from FI to T_j. Crucially, the contribution from each C_ℓ can be charged to at most two different tasks T_j in this way: once for the leftmost T_j that it intersects, and once for the rightmost T_j that it intersects. (The reason is that if some T_j is a subset of C_ℓ, then by definition no step of T_j fails and hence E_j did not occur. So this charging can only occur for a task T_j that intersects C_ℓ but is not a subset of C_ℓ, of which there are at most two.) Moreover, this charging to T_j adds up to a total value of at least $\min \left\{ 1, \sum_{s_i \in T_j} t_i^M \right\}$. Since the charging occurs with probability $1/2$ for each $T_j \in \mathcal{T}_{NI}$, and we are at most double-counting each $\omega(C_\ell)$, we conclude that the expected sum of all $\omega(C_\ell)$, plus t_i^* for all failed tasks s_i that lie in some $T_j \in \mathcal{T}_{NI}$, is at least $1/4$ of $\sum_{T_j \in \mathcal{T}_{NI}} \min \left\{ 1, \sum_{s_i \in T_j} t_i^M \right\}$.

The charging for $(s_i) \in \mathcal{T}_I$ remains unchanged: the value t_i^* is charged in the event that step s_i fails, which occurs with probability at least $1/2$.

Taken together, we obtain our desired 8 approximation. □

We suspect the factor of 8 in Proposition 8 is not tight. But, as we now show, the factor is strictly greater than the factor 4 from Proposition 7, so the assumption that $t_i^M \geq 1$ for all s_i is necessary for Proposition 7 to hold.

Consider a step sequence $\mathcal{S}$ that alternates between (a) steps of manual cost 0 and success probability $1/\sqrt{2} - \epsilon$ where ϵ is vanishingly small, and (b) steps of manual cost 1 and success probability 1. Suppose there are $K > 1$ such pairs. The task sequence described in the proof of

Proposition 8 above groups the steps into pairs of two-step tasks, for a total cost of $K\sqrt{2}$ (ignoring ϵ). The fragmentation index is 1 plus 1 for each task that fails, which is $1 + K(1 - 1/\sqrt{2})$ (again ignoring the impact of ϵ). As K grows large, the ratio tends to $2(\sqrt{2} + 1)$ which is strictly greater than 4.

B Technical Details – Derivation of Effective AI Quality Heterogeneity Distribution

In this appendix we describe how to obtain the analytical formula for distribution of heterogeneity $\phi(\bar{\alpha})$ expressed in (20) by extending the method of Levhari (1968).

Define $u = (w_A\,\tau_A)/(1 - w_M\,\tau_M)$ and rewrite equation (19) as:

$$\left(\int_u^1 \phi(\bar{\alpha})\,d\bar{\alpha}\right)^{\rho} = \theta_A \left(\tau_A \int_u^1 \frac{\phi(\bar{\alpha})}{\bar{\alpha}}\,d\bar{\alpha}\right)^{\rho} + \theta_M \left(\tau_M \int_u^1 \phi(\bar{\alpha})\,d\bar{\alpha}\right)^{\rho} + (1 - \theta_A - \theta_M)\,. \tag{23}$$

Our goal is to solve for $\phi(\bar{\alpha})$ in the equation above as a function of θ_A, θ_M, ρ. Define:

$$\Gamma(u) = \int_u^1 \phi(\bar{\alpha})\,d\bar{\alpha}, \tag{24}$$

and

$$\Psi(u) = \int_u^1 \frac{\phi(\bar{\alpha})}{\bar{\alpha}}\,d\bar{\alpha}. \tag{25}$$

Note that $\Gamma^{'}(u) = -\phi(u)$ and $\Psi^{'}(u) = -\phi(u)/u$. Rewrite equation (23) as:

$$\Gamma^{\rho}(u) = \theta_A\,\tau_A^{\rho}\,\Psi^{\rho}(u) + \theta_M\,\tau_M^{\rho}\,\Gamma^{\rho}(u) + (1 - \theta_A - \theta_M).$$

Rearranging terms, we obtain:

$$\left(1 - \theta_M\,\tau_M^{\rho}\right)\Gamma^{\rho}(u) = \theta_A\,\tau_A^{\rho}\,\Psi^{\rho}(u) + (1 - \theta_A - \theta_M). \tag{26}$$

Differentiating equation (26) with respect to u yields:

$$\left(1 - \theta_M\,\tau_M^{\rho}\right)\Gamma^{\rho-1}(u) = \theta_A\,\tau_A^{\rho}\,\Psi^{\rho-1}(u)\,u^{-1}.$$

Solving for $\Psi^{\rho-1}(u)$ as a function of u and $\Gamma^{\rho-1}(u)$ we get:

$$\Psi^{\rho-1}(u) = \frac{\left(1 - \theta_M\,\tau_M^{\rho}\right)}{\theta_A\,\tau_A^{\rho}}\Gamma^{\rho-1}(u)\,u.$$

Finally, express $\Psi(u)$ in terms of $\Gamma(u)$:

$$\Psi(u) = \left(1 - \theta_M\,\tau_M^{\rho}\right)^{\frac{1}{\rho-1}} \left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}} u^{\frac{1}{\rho-1}}\,\Gamma(u). \tag{27}$$

Substituting $\Psi(u)$ from equation (27) into equation (26), we have:

$$\left(1 - \theta_M\,\tau_M^{\rho}\right)\Gamma^{\rho}(u) = \left(1 - \theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}} \left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}} u^{\frac{\rho}{\rho-1}}\,\Gamma^{\rho}(u) + (1 - \theta_A - \theta_M)\,.$$

Rearranging terms to solve for $\Gamma(u)$, we get:

$$\Gamma(u) = (1 - \theta_A - \theta_M)^{\frac{1}{\rho}} \left[1 - \theta_M\,\tau_M^{\rho} - \left(1 - \theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}} \left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}} u^{\frac{\rho}{\rho-1}}\right]^{-\frac{1}{\rho}}.$$

Recall that $\Gamma'(u) = -\phi(u)$. Differentiating $\Gamma(u)$, we obtain:

$$\begin{aligned}\Gamma'(u) &= \frac{\partial}{\partial u}\left[(1-\theta_A-\theta_M)^{\frac{1}{\rho}}\left[1-\theta_M\,\tau_M^{\rho}-\left(1-\theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}}\left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}}u^{\frac{\rho}{\rho-1}}\right]^{-\frac{1}{\rho}}\right] \\ &= \frac{(1-\theta_A-\theta_M)^{\frac{1}{\rho}}\left(1-\theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}}\left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}}}{\rho-1}u^{\frac{1}{\rho-1}}\left[1-\theta_M\,\tau_M^{\rho}-\left(1-\theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}}\left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}}u^{\frac{\rho}{\rho-1}}\right]^{-\frac{1+\rho}{\rho}}.\end{aligned}$$

Thus, we have:

$$\phi(\bar{\alpha}) = \frac{(1-\theta_A-\theta_M)^{\frac{1}{\rho}}\left(1-\theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}}\left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}}}{1-\rho}(\bar{\alpha})^{\frac{1}{\rho-1}}\left[1-\theta_M\,\tau_M^{\rho}-\left(1-\theta_M\,\tau_M^{\rho}\right)^{\frac{\rho}{\rho-1}}\left(\theta_A\,\tau_A^{\rho}\right)^{\frac{1}{1-\rho}}(\bar{\alpha})^{\frac{\rho}{\rho-1}}\right]^{-\frac{1+\rho}{\rho}}, \tag{28}$$

which is the formula provided in (20) in the main text.

C Appendix Tables

Table C.1: Role of AI-able Task Fragmentation in Determining AI Execution Outcomes (Using Execution-Based Empirical Fragmentation Measures)

	EFI Definition 3			EFI Definition 4		
	(1)	(2)	(3)	(4)	(5)	(6)
AI Exposure	0.43***	0.20***	0.15***	0.26***	0.13***	0.11***
	(0.03)	(0.04)	(0.05)	(0.02)	(0.03)	(0.03)
Empirical Fragmentation Index	-2.58***	-1.97***	-1.68***	-1.48***	-1.29***	-1.24***
	(0.12)	(0.15)	(0.17)	(0.05)	(0.05)	(0.06)
R-squared	0.65	0.74	0.80	0.84	0.87	0.90
R-squared Adj.	0.65	0.74	0.77	0.84	0.87	0.88
N	872	872	872	872	872	872
SOC Group Fixed Effects		Major	Minor		Major	Minor

Clustered standard errors in parentheses. $^{*}: p < 0.1$, $^{**}: p < 0.05$, $^{***}: p < 0.01$.

Notes: This table shows that occupation-level AI execution outcomes depend not only on the overall exposure of tasks to AI, but also on where AI-able steps are situated within the production sequence. The table reports results from estimating regression (21). The dependent variable in all specifications is the share of steps executed by AI in an occupation (ai_execution). The variable "AI Exposure" denotes the share of AI-exposed (E1) steps in the occupation, while the "Empirical Fragmentation Index" captures how dispersed AI-able steps are across the occupation's workflow. Definition 3 measures fragmentation based on realized AI chains per the model's definition (given in Definition 4 of the model section), whereas Definition 4 expands the definition of AI chains by treating both automated and augmented tasks contributing equally to the formation of AI chains in measuring task fragmentation.

Table C.2: Example O*NET Tasks Associated with Claude Conversations, and Calculated Augmentation vs. Automation Outcomes

O*NET Task	**Share of Claude Conversations (%)**								**Label**
	Validation	**Task Iteration**	**Learning**	**Directive**	**Feedback Loop**	**Filtered**	**Augmentation**	**Automation**	
Provide road information to assist motorists.	0.00	11.47	44.39	33.42	5.74	4.99	**55.86**	39.15	Augmentation
Calculate costs of orders, and charge or forward invoices to appropriate accounts.	0.00	0.00	0.00	68.00	0.00	32.00	0.00	**68.00**	Automation
Process data for analysis, using computers.	4.23	18.60	31.40	25.79	16.38	3.59	**54.23**	42.18	Augmentation
Adapt text to accommodate musical requirements of composers and singers.	0.00	40.00	0.00	58.26	0.00	1.74	40.00	**58.26**	Automation
Provide system design and integration recommendations.	2.13	29.75	39.98	21.23	5.80	1.11	**71.87**	27.02	Augmentation
Communicate traffic and crossing rules and other information to students and adults.	0.00	0.00	0.00	65.52	0.00	34.48	0.00	**65.52**	Automation
Prepare, manipulate, and manage extensive databases.	0.00	25.89	22.34	23.40	24.47	3.90	**48.23**	47.87	Augmentation
Prepare, administer, and grade tests and assignments to evaluate students' progress.	7.64	20.94	7.39	58.13	3.94	1.97	35.96	**62.07**	Automation
Conduct statistical analyses to quantify risk using statistical analysis software or econometric models.	0.00	25.23	31.53	17.12	18.92	7.21	**56.76**	36.04	Augmentation
Assemble, typeset, scan and produce digital camera-ready art or film negatives and printer's proofs.	0.00	21.88	0.00	77.34	0.00	0.78	21.88	**77.34**	Automation
Conduct searches to find needed information, using such sources as the internet.	1.31	5.53	58.30	23.14	3.57	8.15	**65.14**	26.71	Augmentation
Compile data and create statistical reports on library usage.	0.00	0.00	0.00	64.86	0.00	35.14	0.00	**64.86**	Automation
Create custom illustrations or other graphic elements.	0.00	48.17	3.17	43.82	2.74	2.10	**51.34**	46.56	Augmentation
Confer with clients to obtain and provide information when claims are made on a policy.	0.00	0.00	45.45	0.00	0.00	54.55	**45.45**	0.00	Augmentation

Notes: Augmentation percentage is the sum of Validation, Task Iteration, and Learning types of conversations whereas the automation percentage is the sum of Directive and Feedback Loop types of conversations. Whichever mode between Augmentation and Automation has a higher share is selected as the task's mode of AI execution, ignoring the Filtered share.

Table C.3: Example DWAs with Tasks in Multiple Occupations and the Tasks Surviving the "Finding Similar Tasks" Procedure

Panel (A)

O*NET DWA Title	O*NET Task Title	O*NET Occupation Title	Execution Label
Analyze operational or research data.	Analyze or manipulate bioinformatics data using software packages, statistical applications, or data mining techniques.	Bioinformatics Technicians	Augmentation
	Conduct quality analyses of data inputs and resulting analyses or predictions.		Augmentation
	Analyze research data to determine its significance, using computers.	Astronomers	Augmentation
	Collect and analyze data, such as studying old records, tallying the number of outpatients entering each day or week, or participating in federal, state, or local population surveys as a Census Enumerator.	Interviewers, Except Eligibility and Loan	Manual
	Analyze data to determine answers to questions from customers or members of the public.	Receptionists and Information Clerks	Augmentation
	Compute and analyze data, using statistical formulas and computers or calculators.	Statistical Assistants	Augmentation
Prepare research or technical reports on environmental issues.	Write reports or articles for Web sites or newsletters related to environmental engineering issues.	Environmental Engineers	Manual
	Prepare technical and research reports, such as environmental impact reports, and communicate the results to individuals in industry, government, or the general public.	Biologists	Manual
	Prepare scientific atmospheric or climate reports, articles, or texts.	Atmospheric and Space Scientists	Automation
	Prepare charts or graphs from data samples, providing summary information on the environmental relevance of the data.	Environmental Scientists and Specialists, Including Health	Augmentation
	Write reports or academic papers to communicate findings of climate-related studies.	Climate Change Policy Analysts	Manual
	Prepare study reports, memoranda, briefs, testimonies, or other written materials to inform government or environmental groups on environmental issues, such as climate change.		Manual
	Prepare technical and research reports, such as environmental impact reports, and communicate the results to individuals in industry, government, or the general public.	Industrial Ecologists	Manual
	Produce environmental documents, such as environmental assessments or environmental impact statements.	Transportation Planners	Manual
Document events or evidence using photographic or audiovisual equipment.	Take photographs and motion pictures for use in lectures and publications and to develop displays.	Park Naturalists	Manual
	Create data records for use in describing and analyzing social patterns and processes, using photography, videography, and audio recordings.	Anthropologists and Archeologists	Automation
	Create photographic recordings of information, using equipment.	Geological Technicians, Except Hydrologic Technicians	Automation
	Use photographic or video equipment to document evidence or crime scenes.	Forensic Science Technicians	Manual

Panel (B)

O*NET DWA Title	O*NET Task Title	O*NET Occupation Title	Execution Label
Analyze operational or research data.	Analyze or manipulate bioinformatics data using software packages, statistical applications, or data mining techniques.	Bioinformatics Technicians	Augmentation
	Analyze research data to determine its significance, using computers.	Astronomers	Augmentation
	Compute and analyze data, using statistical formulas and computers or calculators.	Statistical Assistants	Augmentation
Prepare research or technical reports on environmental issues.	Prepare technical and research reports, such as environmental impact reports, and communicate the results to individuals in industry, government, or the general public.	Biologists	Manual
	Prepare scientific atmospheric or climate reports, articles, or texts.	Atmospheric and Space Scientists	Automation
	Prepare charts or graphs from data samples, providing summary information on the environmental relevance of the data.	Environmental Scientists and Specialists, Including Health	Augmentation
	Prepare study reports, memoranda, briefs, testimonies, or other written materials to inform government or environmental groups on environmental issues, such as climate change.	Climate Change Policy Analysts	Manual
	Prepare technical and research reports, such as environmental impact reports, and communicate the results to individuals in industry, government, or the general public.	Industrial Ecologists	Manual
	Produce environmental documents, such as environmental assessments or environmental impact statements.	Transportation Planners	Manual
Document events or evidence using photographic or audiovisual equipment.	Take photographs and motion pictures for use in lectures and publications and to develop displays.	Park Naturalists	Manual
	Create data records for use in describing and analyzing social patterns and processes, using photography, videography, and audio recordings.	Anthropologists and Archeologists	Automation
	Create photographic recordings of information, using equipment.	Geological Technicians, Except Hydrologic Technicians	Automation

Notes: Panel (A) shows three example detailed work activities (DWAs) and tasks associated with them in the O*NET dataset. Entries highlighted with a red font are those that are dropped in the "finding similar tasks" procedure described in text. Panel (B) shows the set of tasks for DWAs of Panel (A) that successfully survive the "finding similar tasks" procedure.

Table C.6: Effect of Neighboring Tasks' AI Execution Status on Task's AI Execution Likelihood (GPT-filtered Sample)

Specification	(1)	(2)	(3)	(4)	(5)	(6)
$(t-2)$ Task AI	0.04	0.01	0.01	0.04	0.04	0.04
	(0.03)	(0.02)	(0.02)	(0.03)	(0.04)	(0.03)
$(t-1)$ Task AI	0.11***	0.07**	0.06**	0.06**	0.10***	0.06**
	(0.04)	(0.03)	(0.02)	(0.03)	(0.04)	(0.03)
$(t+1)$ Task AI	0.11***	0.08***	0.07***	0.09***	0.09***	0.09***
	(0.03)	(0.03)	(0.03)	(0.03)	(0.03)	(0.03)
$(t+2)$ Task AI	0.01	-0.02	-0.02	-0.00	0.00	-0.00
	(0.03)	(0.02)	(0.02)	(0.03)	(0.04)	(0.03)
Pseudo R^2	0.031	0.055	0.065	0.198	0.020	0.198
Observations	3,689	3,689	3,567	2,544	2,544	2,544
SOC Group FE		Major	Minor			
DWA FE				Yes		Yes
NumTasks in DWA-Occupation Control					Yes	Yes

Clustered standard errors in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Notes: This table reports average marginal effects from estimating regression (22). The dependent variable in all specifications is an indicator for whether task t is AI-executed (is_ai_t). The sample is restricted to similar tasks across occupations, identified by GPT-5-mini as tasks belonging to the same DWA with similar execution nature and skill characteristics. All specifications control for the AI exposure status of task t and for the total number of tasks in task t's occupation. Standard errors are bootstrapped using $B = 200$ replications and clustered at the DWA level.

Table C.7: Effect of Neighboring Tasks' AI Execution Status on Task's AI Automation Likelihood

Specification	(1)	(2)	(3)	(4)	(5)	(6)
$(t-2)$ Task AI	0.02**	-0.01	-0.01	-0.01	0.01	-0.01
	(0.01)	(0.01)	(0.01)	(0.02)	(0.03)	(0.02)
$(t-1)$ Task AI	0.05***	0.02*	0.01	0.02	0.09**	0.02
	(0.02)	(0.01)	(0.01)	(0.03)	(0.04)	(0.03)
$(t+1)$ Task AI	0.05***	0.02***	0.02**	0.01	0.09**	0.01
	(0.01)	(0.01)	(0.01)	(0.02)	(0.03)	(0.02)
$(t+2)$ Task AI	0.03***	0.00	-0.00	0.02	0.05*	0.02
	(0.01)	(0.01)	(0.01)	(0.02)	(0.03)	(0.02)
Pseudo R^2	0.101	0.192	0.173	0.245	0.033	0.245
Observations	13,786	12,777	10,390	2,815	2,815	2,815
SOC Group FE		Major	Minor			
DWA FE				Yes		Yes
NumTasks in DWA-Occupation Control					Yes	Yes

Clustered standard errors in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Notes: This table reports average marginal effects from estimating regression (22) but with dependent variable (is_automated_t) instead of (is_ai_t). The sample is restricted to similar tasks across occupations, identified as tasks belonging to the same DWA in the O*NET dataset. Tasks that are mapped to multiple DWAs are retained, with one associated DWA assigned randomly for estimation to preserve sufficient variation in the data. All specifications control for the AI exposure status of task t and for the total number of tasks in task t's occupation. Standard errors are bootstrapped using $B = 200$ replications and clustered at the DWA level.

Table C.8: Effect of Neighboring Tasks' AI Execution Status on Task's AI Automation Likelihood (GPT-filtered Sample)

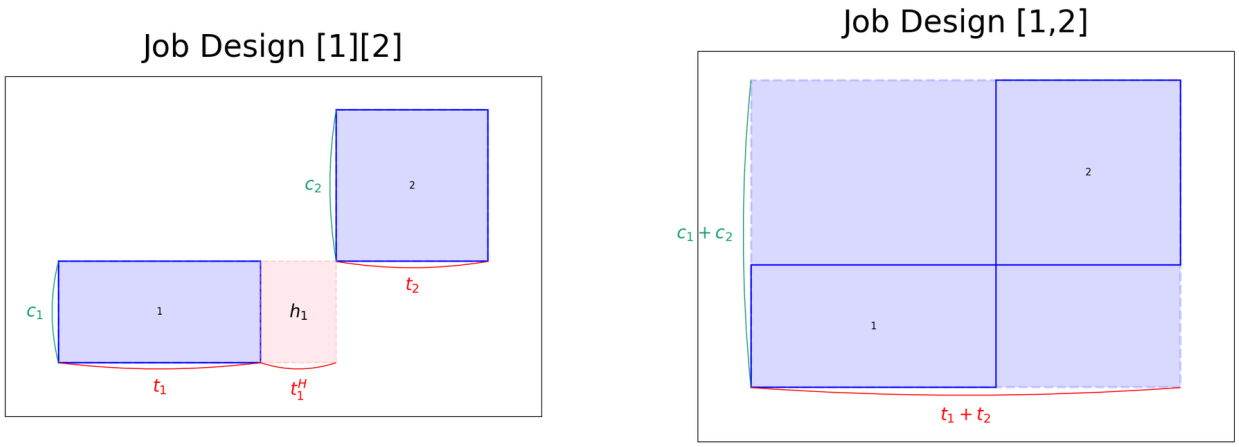
Specification	(1)	(2)	(3)	(4)	(5)	(6)
$(t-2)$ Task AI	0.01	-0.02	-0.02	0.00	0.00	0.00
	(0.01)	(0.01)	(0.01)	(0.02)	(0.03)	(0.02)
$(t-1)$ Task AI	0.06**	0.02	0.02	0.03	0.11**	0.03
	(0.03)	(0.01)	(0.01)	(0.03)	(0.05)	(0.03)
$(t+1)$ Task AI	0.06**	0.03*	0.03	0.02	0.09*	0.02
	(0.03)	(0.02)	(0.02)	(0.03)	(0.05)	(0.03)
$(t+2)$ Task AI	0.01	-0.02	-0.02	0.00	0.02	0.01
	(0.02)	(0.01)	(0.01)	(0.02)	(0.04)	(0.02)
Pseudo R^2	0.042	0.125	0.118	0.247	0.031	0.247
Observations	5,156	5,077	4,490	1,771	1,771	1,771
SOC Group FE		Major	Minor			
DWA FE				Yes		Yes
NumTasks in DWA-Occupation Control					Yes	Yes

Clustered standard errors in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Notes: This table reports average marginal effects from estimating regression (22) but with dependent variable (is_automated_t) instead of (is_ai_t). The sample is restricted to similar tasks across occupations, identified by GPT-5-mini as tasks belonging to the same DWA with similar execution nature and skill characteristics. Tasks that are mapped to multiple DWAs are retained, with one associated DWA assigned randomly for estimation to preserve sufficient variation in the data. All specifications control for the AI exposure status of task t and for the total number of tasks in task t's occupation. Standard errors are bootstrapped using $B = 200$ replications and clustered at the DWA level.

D Appendix Figures

Figure D.1: Illustration of the Trade-off between Worker Specialization and Coordination in Task Assignment



Notes: The blue bounded rectangles indicate tasks. The height of each rectangle corresponds to the task's skill requirement (c) and its width to the time requirement (t). The shaded areas represent the wage bills of jobs. In the left panel, tasks are assigned to two specialized workers but a hand-off cost (the pink rectangle) is introduced due to coordination frictions between them. In the right panel, tasks are bundled into a single job performed by one worker, eliminating hand-off costs but requiring a worker skilled enough to perform both tasks and compensated with a higher per-unit-time wage rate.

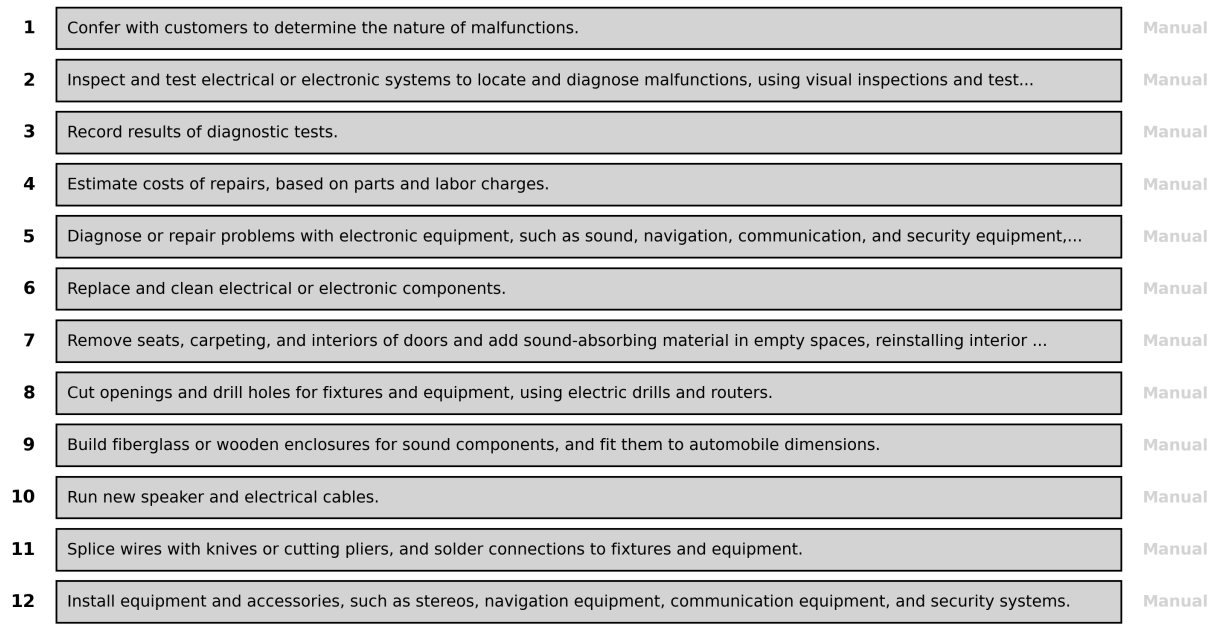
Figure D.2: Task Sequence of Computer Programmers Occupation

Notes: This figure shows the task sequence for an occupation with a high share of its tasks (about two thirds) executed by AI. The task ordering is generated by GPT-5-mini. The execution labels come from the Anthropic Economic Index. The O*NET code for this occupation is 15-1251.

Figure D.3: Task Sequence of Public Relations Specialists Occupation

Notes: This figure shows the task sequence for an occupation with a moderate share of its tasks (about one third) executed by AI. The task ordering is generated by GPT-5-mini. The execution labels come from the Anthropic Economic Index. The O*NET code for this occupation is 27-3031.

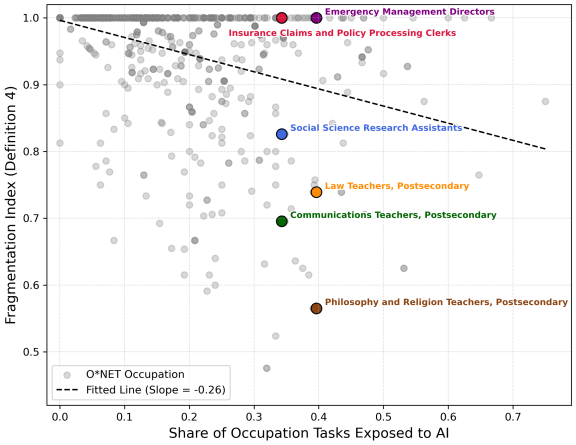
Figure D.4: Task Sequence of Electronic Equipment Installers and Repairers, Motor Vehicles



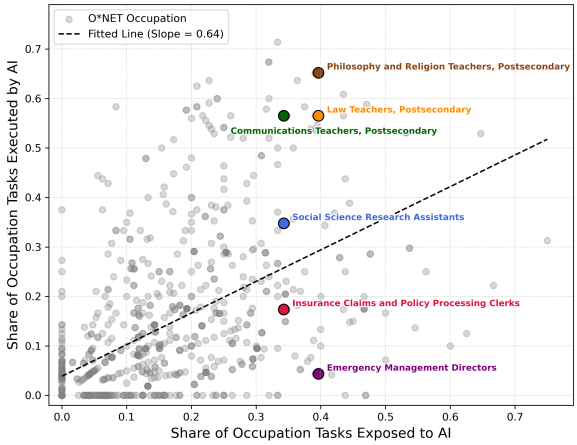
Notes: This figure shows the task sequence for an occupation with none of its tasks executed by AI. The task ordering is generated by GPT-5-mini. The execution labels come from the Anthropic Economic Index. The O*NET code for this occupation is 49-2096.

Figure D.5: Relationships Between Occupational AI Exposure, Empirical Fragmentation, and AI Execution (Execution-Based Empirical Fragmentation Measure)

(a) Empirical Fragmentation Index vs. AI Exposure

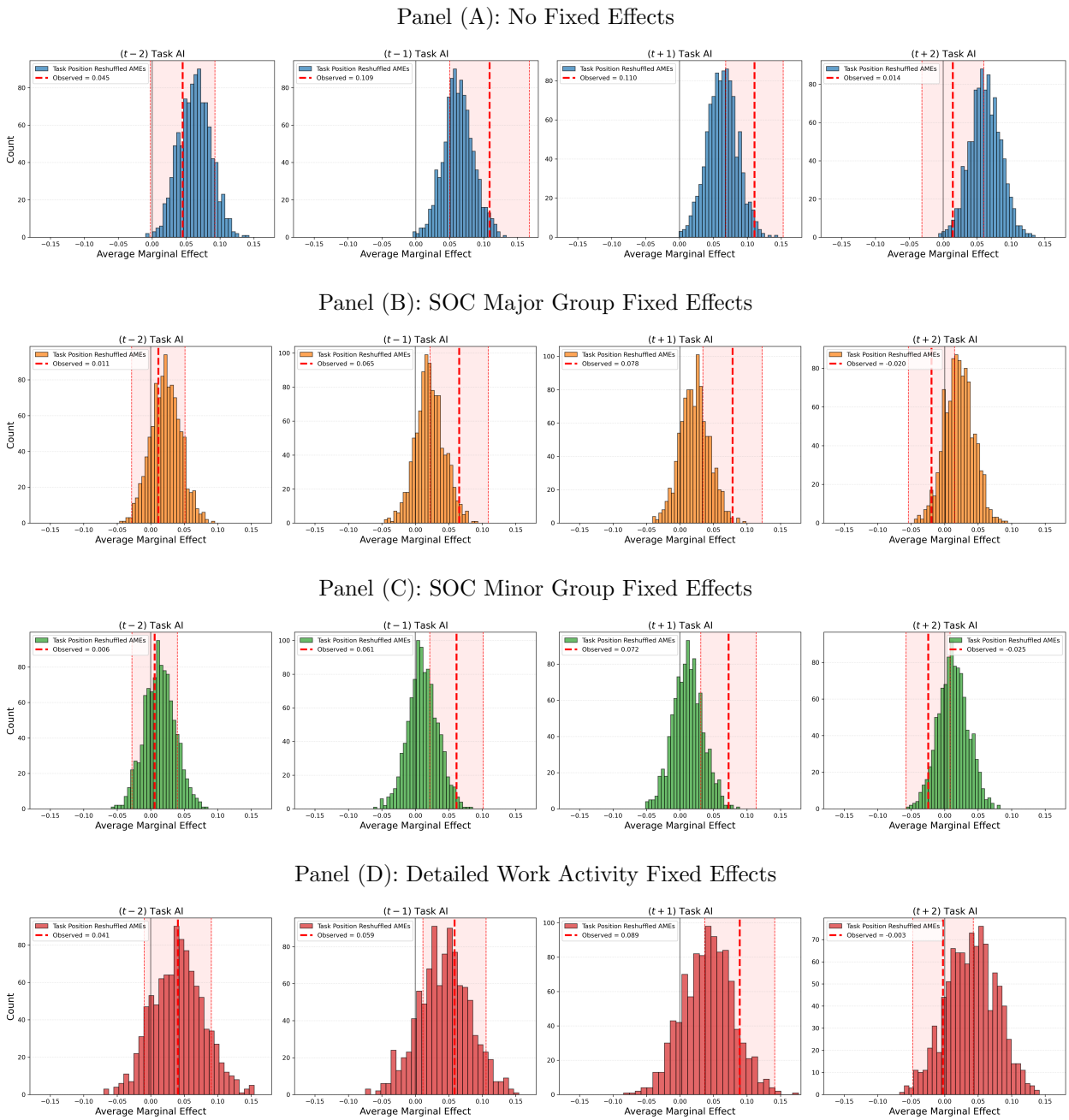


(b) AI Execution vs. AI Exposure



Notes: These graphs show how the fragmentation of AI-able tasks within an occupation's workflow shapes the conversion of AI-exposed tasks into AI-executed tasks. Each bin represents a single O*NET occupation in both panels. Panel (a) plots version 4 of the empirical fragmentation index defined in Table C.1 against the share of AI-exposed tasks, whereas Panel (b) plots the share of realized AI-executed tasks against the share of AI-exposed tasks. All six highlighted occupations have 23 tasks in their production sequence and share similar occupation-level exposure to AI: *Emergency Management Directors, Law Teachers,* and *Philosophy and Religion Teachers* each have 39% of their tasks exposed to AI, whereas *Insurance Claims and Policy Processing Clerks, Social Science Research Assistants,* and *Communications Teachers* each have 35% of their tasks exposed to AI.

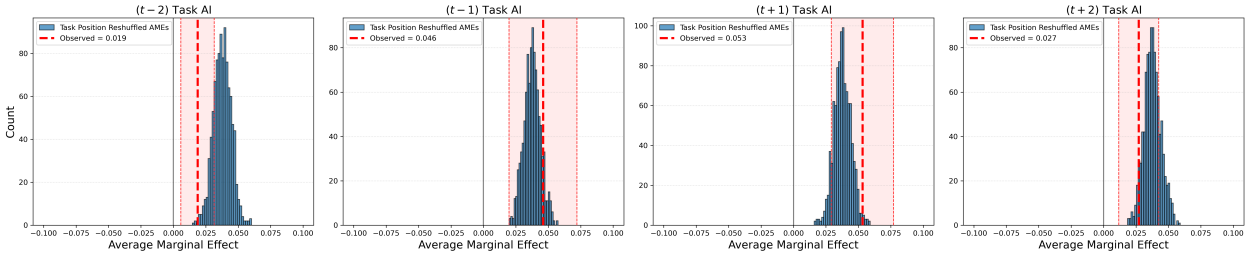
Figure D.6: Effect of Neighboring Tasks' AI Execution Status on Task's AI Execution Likelihood (GPT-filtered Sample)



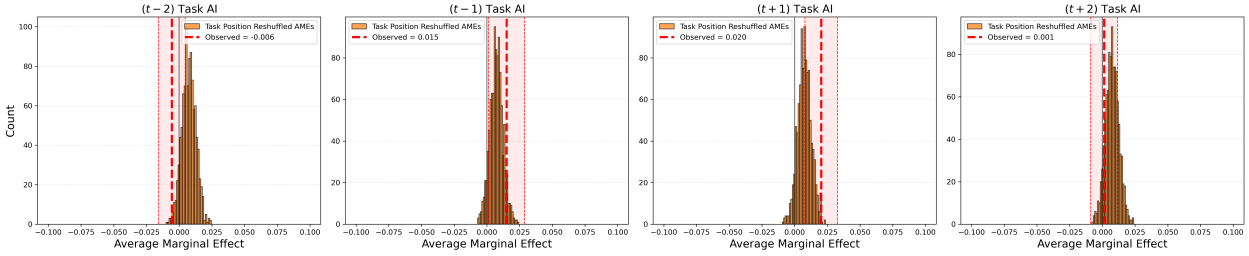
Notes: These graphs show that, among similar tasks appearing in multiple occupations, those whose immediate neighbors are AI-executed exhibit higher probabilities of being executed by AI, while more distant neighbors have little to no effect on a task's AI execution likelihood. The red dashed line in each graph indicates the observed average marginal effect of the corresponding variable in the original dataset reported in Appendix Table C.6. The red shaded area marks the 90% confidence interval around the observed point estimates in the GPT-filtered sample. The histograms plot the distribution of average marginal effects across 1,000 randomized reshuffles of task positions within occupations. Panel (A) reports results from specification (22), while Panels (B), (C), and (D) augment the baseline specification with SOC major group, SOC minor group, and DWA fixed effects, respectively.

Figure D.7: Effect of Neighboring Tasks' AI Execution Status on Task's AI Automation Likelihood

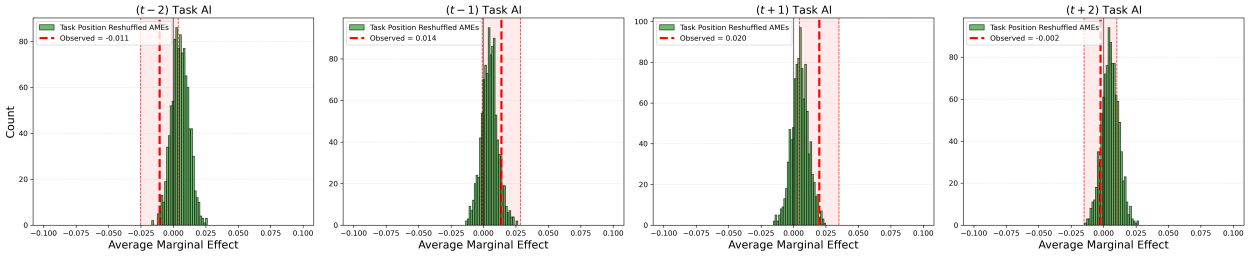
Panel (A): No Fixed Effects



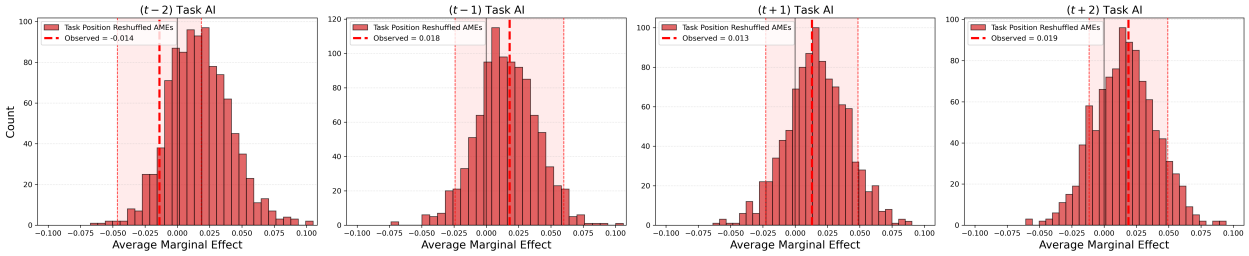
Panel (B): SOC Major Group Fixed Effects



Panel (C): SOC Minor Group Fixed Effects



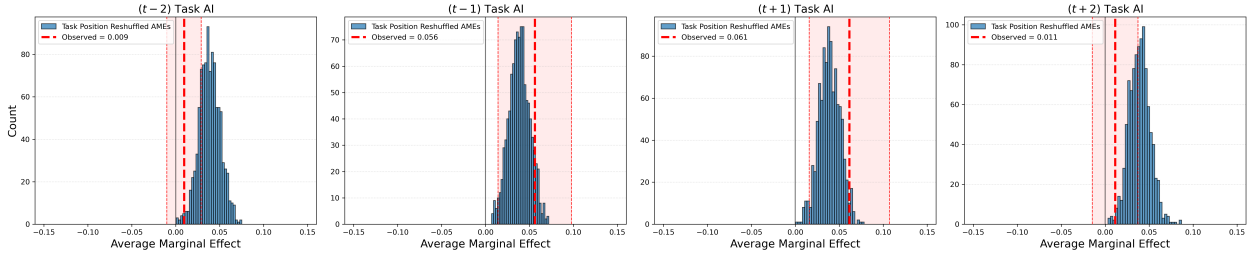
Panel (D): Detailed Work Activity Fixed Effects



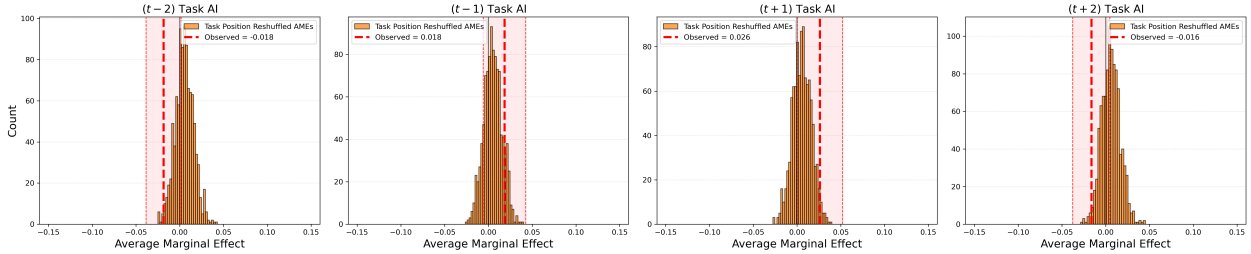
Notes: These graphs show that, among similar tasks appearing in multiple occupations, those whose immediate neighbors are AI-executed exhibit higher probabilities of being automated by AI, while more distant neighbors have little to no effect on a task's AI automation likelihood. The red dashed line in each graph indicates the observed average marginal effect of the corresponding variable in the original dataset reported in Table C.7. The red shaded area marks the 90% confidence interval around the observed point estimates in the main sample. The histograms plot the distribution of average marginal effects across 1,000 randomized reshuffles of task positions within occupations. Panel (A) reports results from specification (22) but with dependent variable (is_automated_t) instead of (is_ai_t), while Panels (B), (C), and (D) augment the baseline specification with SOC major group, SOC minor group, and DWA fixed effects, respectively.

Figure D.8: Effect of Neighboring Tasks' AI Execution Status on Task's AI Automation Likelihood (GPT-filtered Sample)

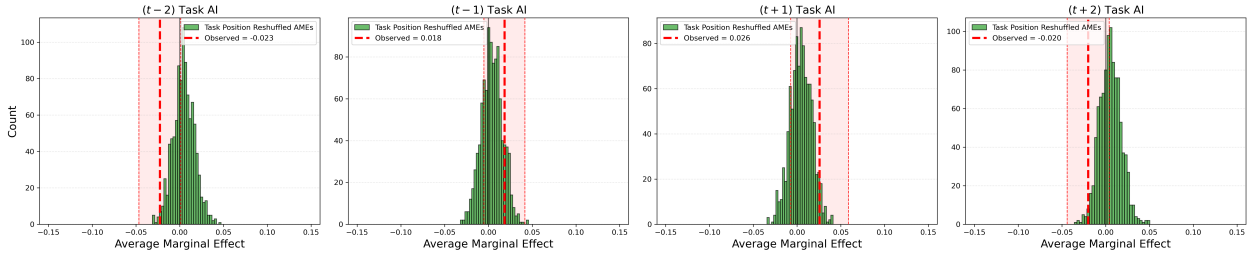
Panel (A): No Fixed Effects



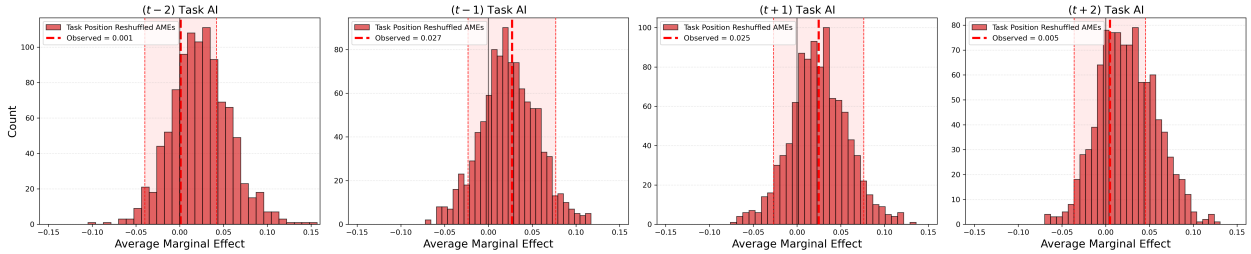
Panel (B): SOC Major Group Fixed Effects



Panel (C): SOC Minor Group Fixed Effects



Panel (D): Detailed Work Activity Fixed Effects



Notes: These graphs show that, among similar tasks appearing in multiple occupations, those whose immediate neighbors are AI-executed exhibit higher probabilities of being automated by AI, while more distant neighbors have little to no effect on a task's AI automation likelihood. The red dashed line in each graph indicates the observed average marginal effect of the corresponding variable in the original dataset reported in Table C.8. The red shaded area marks the 90% confidence interval around the observed point estimates in the main sample. The histograms plot the distribution of average marginal effects across 1,000 randomized reshuffles of task positions within occupations. Panel (A) reports results from specification (22) but with dependent variable (is_automated$_t$) instead of (is_ai$_t$), while Panels (B), (C), and (D) augment the baseline specification with SOC major group, SOC minor group, and DWA fixed effects, respectively.

E GPT-5-mini Prompts

This Appendix provides the GPT-5-mini prompts used to generate a task sequence for each occupation (Prompt 1), and for finding similar tasks across occupations for each DWA (Prompt 2).

Prompt #1: GPT-5-mini Prompt for Ordering of Tasks in Occupation

```
You are an expert in workflow analysis for the occupation: {{ occupation }}.
Below is a list of {{ num_tasks }} tasks that are part of this occupation:
{{ tasks_list }}

Provide the typical sequential order in which these tasks are performed in a
real-world workflow.

Return your answer as a JSON array where each element has:
- "Task Position": the sequence number (1, 2, 3, etc.).
- "Task Title": the exact task text from the list above.
Format: [{"Task Position": 1, "Task Title": "..."},
         {"Task Position": 2, "Task Title": "..."},
         ...]
Only return the JSON array, nothing else.
```

Prompt #2: GPT-5-mini Prompt for Finding Similar Tasks in Each DWA

```
You are an expert in workflow analysis for the detailed work activity:
{{ detailed_work_activity }}.

Below is a list of {{ num_tasks }} task IDs and titles that belong to this
detailed work activity and appear across similar or different occupations
(tasks and occupations are ordered such that the first task belongs to the
first occupation, the second task belongs to the second occupation, etc.).
Tasks IDs: {{ tasks_ids }}
Tasks list: {{ tasks_list }}
Occupations list: {{ occupations_list }}
Occupation Codes list: {{ occupation_codes_list }}

Determine which tasks are similar in nature and in terms of their objectives,
methods, or required skills. There may be more than one task associated with
an occupation. Return only the most relevant task for every occupation.
Only look for tasks that are actually similar. Do not feel obliged to return
all occupations.

Return the task-occupation pairs you determine as similar as a JSON array
where each element has:
- "Task ID": the exact task ID from the list of task IDs above
- "Task Title": the exact task text from the list of tasks above
- "O*NET-SOC Code": the exact occupation code text from the list of occupation
   codes above
- "Occupation Title": the exact occupation text from the list of occupations
   above
Format: [{"Task ID": 1234, "Task Title": "...",
          "O*NET-SOC Code": "...", "Occupation Title": "..."},
          {"Task ID": 5678, "Task Title": "...",
          "O*NET-SOC Code": "...", "Occupation Title": "..."},
          ...]
Only return the JSON array, nothing else.
```

F Data Appendix – Robustness to Alternative GPT Prompts

In this appendix, we assess whether our results are sensitive to the specific GPT prompt used to order tasks within occupations. Although alternative prompts can generate different task sequences, we show that these differences do not exhibit systematic patterns across subsets of occupations, and that our headline results are robust to prompt formulation.

We begin by generating task orderings for each occupation using 10 alternative prompts in addition to the baseline prompt. We use the same structure as the first prompt in Appendix E, but only change the sentence starting with "*Provide the typical sequential...*" with an alternative from the list below:

1. *Imagine a typical workday for this occupation. As the day unfolds, tasks arise and are completed as needed. Order the tasks in the sequence they most naturally occur.*
2. *For each task, consider its inputs and outputs. Order tasks so outputs of earlier tasks plausibly feed into later tasks. If tasks are parallel, place the more upstream task first.*
3. *Order tasks to minimize rework, waiting, and unnecessary handoffs. Assume an experienced worker executing the workflow efficiently.*
4. *Think about what must ultimately be produced in this occupation and what needs to happen before that. Use this reasoning to produce a natural forward sequence of tasks.*
5. *Identify which tasks logically depend on others, then order the tasks in a single sequence consistent with those dependencies and typical practice.*
6. *Order tasks according to how information is generated, transformed, and used over the course of the work.*
7. *Order tasks based on when mistakes would be most costly, placing tasks that prevent or constrain downstream errors earlier.*
8. *Order tasks so that tasks informing important decisions tend to occur before tasks that rely on those decisions.*
9. *Order the tasks as an experienced practitioner would intuitively carry them out, without explicitly planning or formalizing the workflow.*
10. *Order the tasks to reflect how the work is most commonly carried out in practice, rather than how it is formally described.*

Notice that these prompts are not simple re-wordings of the same instruction. Instead, they approach the task ordering problem from different perspectives, so that, in principle, the resulting task sequences could differ. Our aim is to quantify how much variation these alternative prompt formulations induce in practice, and whether such variation poses a concern for our purposes.

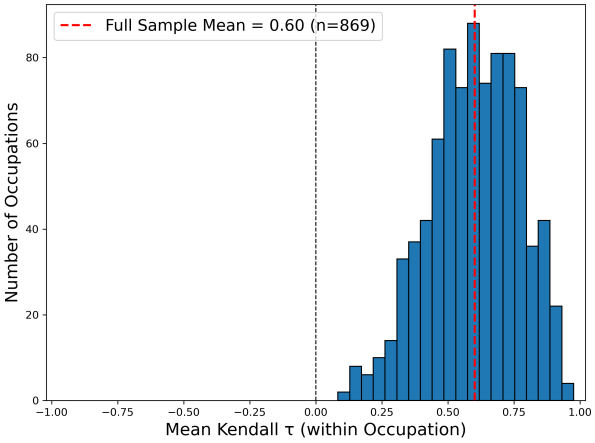
To measure the degree of overlap between task orderings, we compute pairwise Kendall's τ within each occupation across all pairs of prompts.[32] Across all occupations and prompt pairs, the average Kendall's τ is 0.6, with the full distribution shown in Panel (a) of Figure F.1. This value

[32]Kendall's τ is a standard rank correlation measure that, roughly speaking, captures the fraction of task pairs that appear in the same relative order across two ranked lists. This measure ranges from -1 (completely reversed orderings) to 1 (identical orderings), with 0 indicating no systematic overlap. Higher values therefore correspond to greater similarity between task sequences.

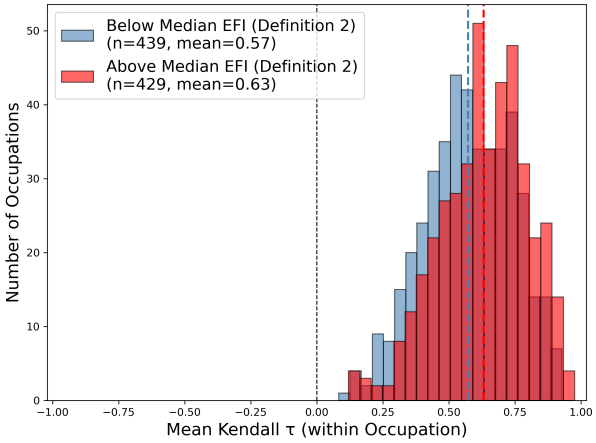
implies that task orderings are not identical across prompts, but that there is a reasonably high degree of overlap given the diversity of prompt formulations considered. More importantly for our analyses, we find no evidence that GPT generates systematically different task orderings across different types of occupations. Specifically, we split occupations based on whether they fall above or below the median in the *main prompt sample* along three dimensions discussed in Subsection 7.2: the empirical fragmentation index (Definition 2), the share of occupation tasks exposed to AI (E1), and the share of tasks executed by AI. Panels (b)–(d) of Figure F.1 show that the mean Kendall's τ is very similar for above- and below-median occupations in all cases.

Figure F.1: Kendall's τ Distribution Across GPT Prompt Task Orderings

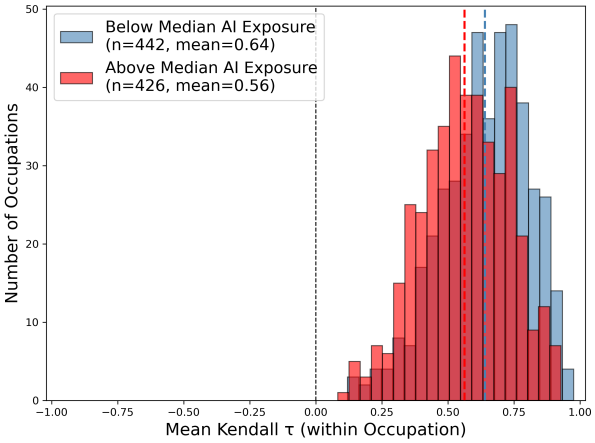
(a) Full Sample



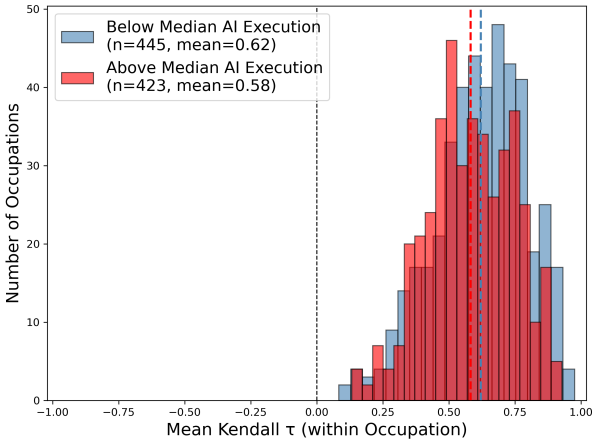
(b) Empirical Fragmentation Index Split



(c) AI Exposure (E1) Split



(d) AI Execution Split



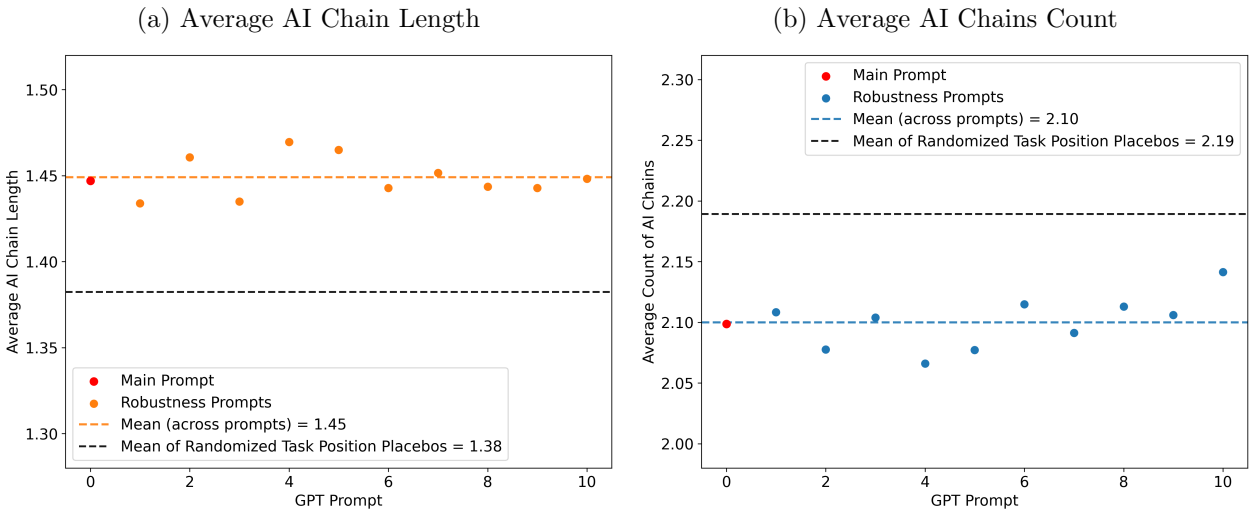
Notes: Each panel shows the distribution of mean Kendall's τ computed across 11 prompts (main + ten alternatives) within occupations. Split figures in Panels (b)–(d) compare occupations above (red) and below (blue) the median of the indicated measure, computed in the sample generated by the main prompt.

In the following Subsections, we show that we obtain the same results for each of our headline analyses using these alternative prompts.

F.1 Robustness of Prediction #1 Results to Alternative GPT Prompts

Here, we examine how sensitive the AI chain length and AI chains count statistics are to the choice of GPT prompt. Panel (a) of Figure F.2 reports the average AI chain length computed under each prompt, and Panel (b) reports the corresponding average AI chains count. Across prompts, both statistics remain tightly clustered around the baseline value implied by the main prompt. In particular, the mean across alternative prompts (the dashed colored lines) matches the main sample estimate from the main prompt, which lies in the extreme tail of the placebo distributions in Figure 8 in both cases.

Figure F.2: Robustness of AI Chain Length and AI Chain Counts to GPT Prompts



Notes: This figure shows that the average AI chain length and average AI chains count measures are robust to alternative prompt formulations. Panel (a) reports average AI chain length, and Panel (b) reports the average number of AI chains per occupation, computed separately under each prompt. Prompt 0 corresponds to the baseline prompt used in the main text. The dashed black reference line in each panel is the mean of the same statistic across 1,000 randomized task-position reshuffle placebos shown in Figure 8. The dashed colored line reports the mean of the statistic across all 11 prompts.

F.2 Robustness of Prediction #2 Results to Alternative GPT Prompts

Here, we re-estimate Equation (21) using task orderings generated by the alternative prompts and report the analogs of the estimates in Tables 3 and C.1.

Figure F.3 plots the equivalents of estimated coefficients reported in Table 3 for the main variables across the three specifications (baseline, SOC major group fixed effects, and SOC minor group fixed effects) for empirical fragmentation index Definition 1 in Panel (A) and Definition 2 in Panel (B). Across all specifications, the estimates obtained under alternative prompts fall within the range of the main prompt estimates in both magnitude and statistical significance. Specifically, the estimates on occupation-level AI exposure are positive and statistically different from zero throughout. For EFI, the Definition 1 coefficients in Panel (A) vary in sign but remain statistically indistinguishable from zero, while the Definition 2 coefficients in Panel (B) are consistently negative

and statistically different from zero. This mirrors the results obtained under the main prompt and reflects measurement differences across EFI definitions. Recall that under Definition 1 only about 14% of tasks are labeled AI-exposed under the E1 measure, which makes AI chain formation sparse and leaves EFI weakly measured. Definition 2, however, allows both E1 and E2 exposure labels to form AI chains, yielding a denser exposure signal and therefore a more precisely estimated relationship between fragmentation and AI execution.

Similarly, Figure F.4 plots the equivalents of estimates reported in Table C.1 for the two ex-post, execution-based EFI definitions 3 and 4 using alternative prompts. Again, similar to the results obtained from the main prompt, the estimated coefficient on AI exposure is positive and the estimated coefficient on empirical fragmentation index is negative.

F.3 Robustness of Prediction #3 Results to Alternative GPT Prompts

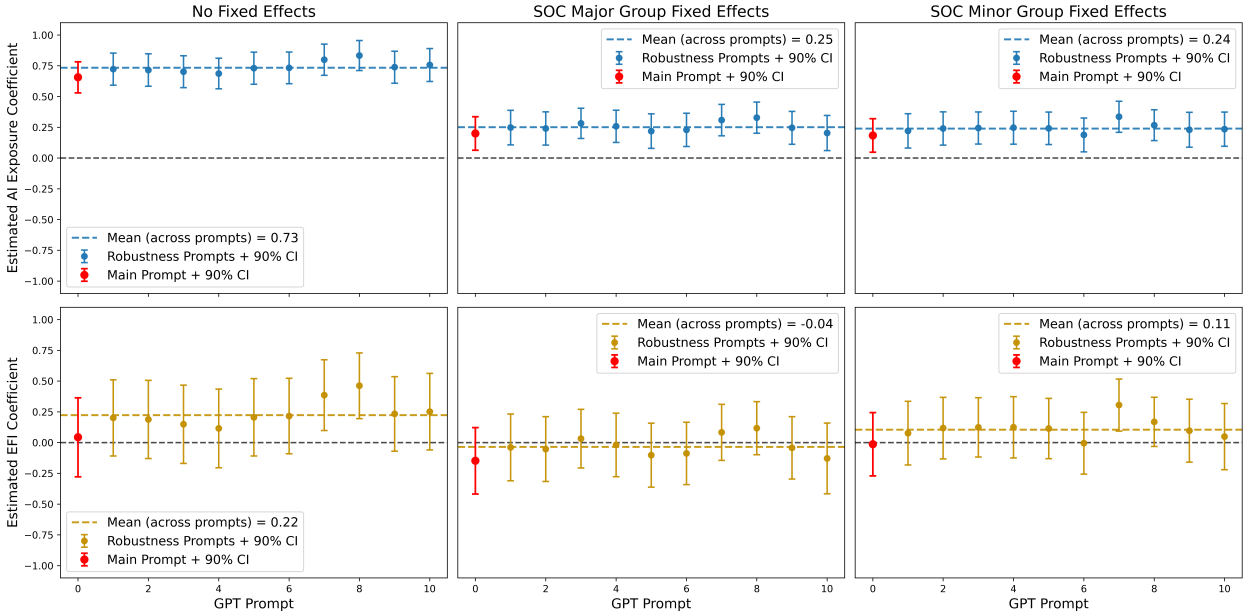
Here, we assess the robustness of our spillover results for neighboring tasks being AI-executed on the focal task's likelihood of AI execution.

For each alternative prompt, we re-estimate Equation (22) on the corresponding main sample under four specifications: baseline, SOC major group fixed effects, SOC minor group fixed effects, and DWA fixed effects. Figure F.5 reports the equivalent of average marginal effects reported in columns (1)–(4) of Table 4 for all prompts. The layout and color coding of specifications in this figure mirror those in Figure 10 from the reshuffled task positions robustness exercise.

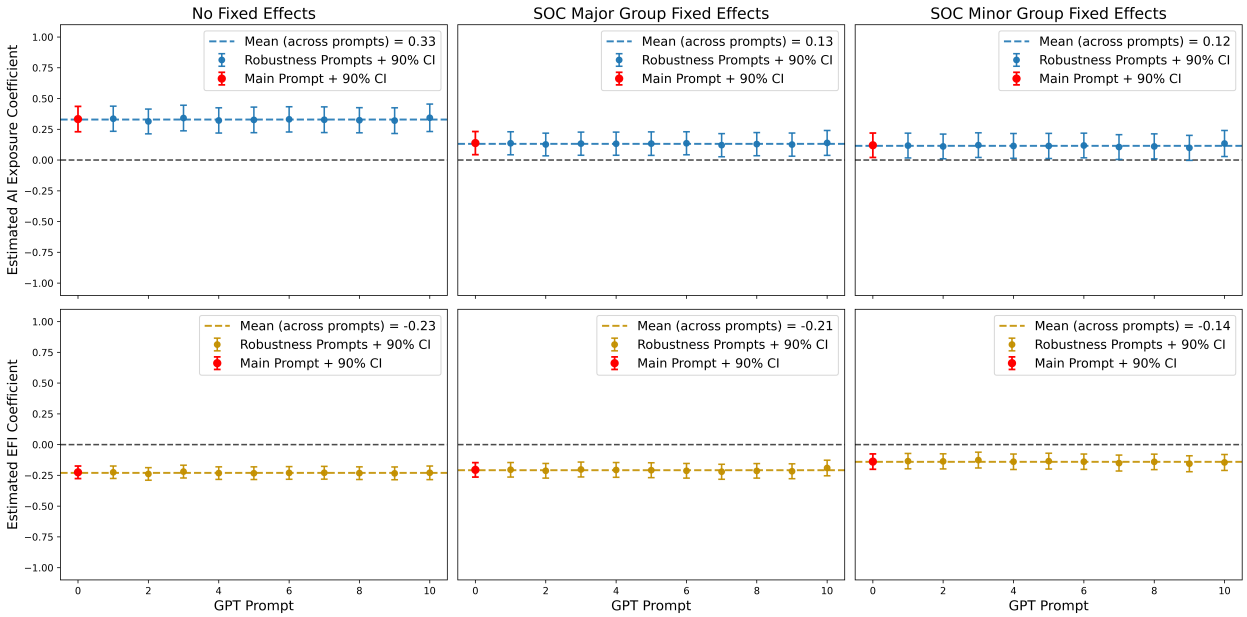
We find that even though the average effect of more distant neighbors across alternative prompts is slightly larger than under the main prompt, the substantive conclusion remains unchanged. The two immediate neighbors (middle columns) still have larger and more precisely estimated effects than more distant neighbors (side columns), and the effects of distant neighbors attenuate and become less distinguishable from zero as additional fixed effects are included in Panels (B)–(D).

Figure F.3: Robustness of Table 3 Estimates to GPT Prompts

Panel (A): Empirical Fragmentation Index Definition 1



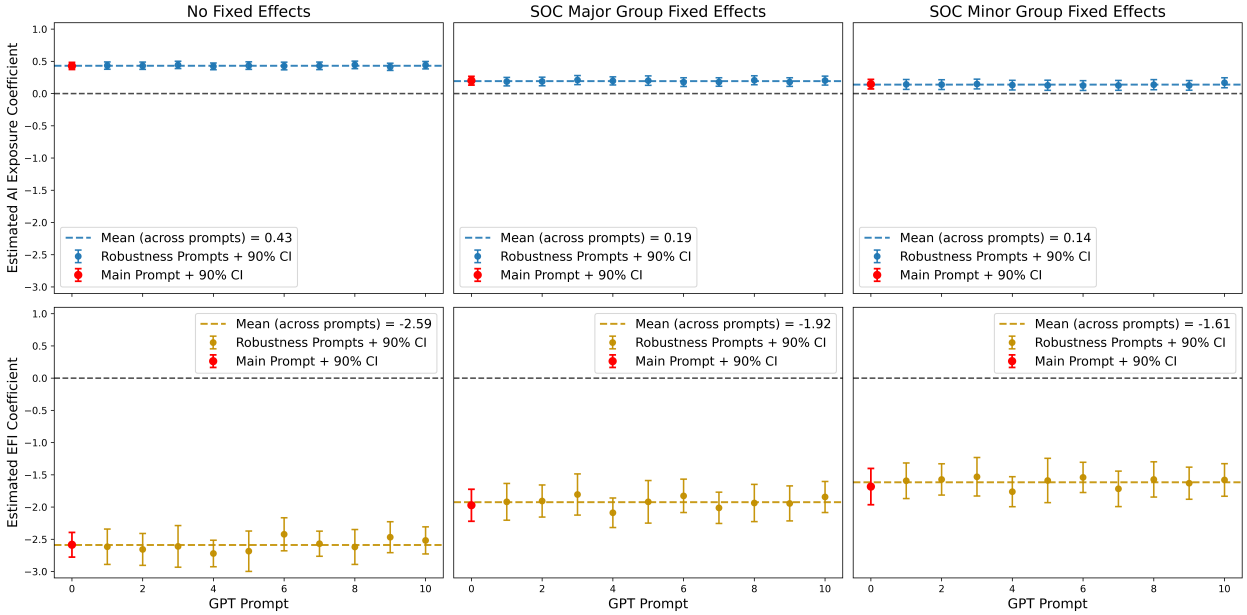
Panel (B): Empirical Fragmentation Index Definition 2



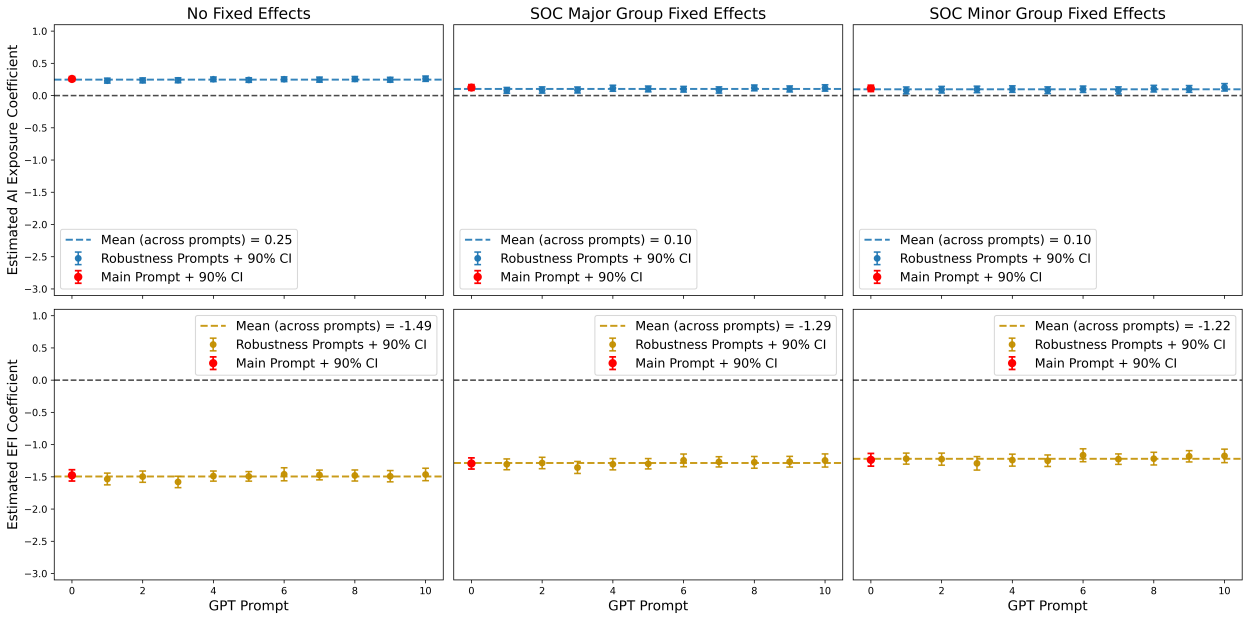
Notes: This figure shows that the estimated relationships between occupation-level AI execution and both AI exposure and the empirical fragmentation index are robust to alternative prompt formulations. Panel (A) reports coefficients on AI exposure (top row, blue) and EFI (bottom row, gold) for exposure-based EFI Definition 1. Panel (B) reports the corresponding coefficients for exposure-based EFI Definition 2. Error bars denote 90% confidence intervals. The dashed colored lines report the mean across all 11 prompts. In each graph, the Prompt 0 estimate is highlighted in red and corresponds to the estimate reported in Table 3 using the main prompt. The Panel (A) graphs correspond to columns (1)–(3), and the Panel (B) graphs correspond to columns (4)–(6) of Table 3.

Figure F.4: Robustness of Table C.1 Estimates to GPT Prompts

Panel (A): Empirical Fragmentation Index Definition 3



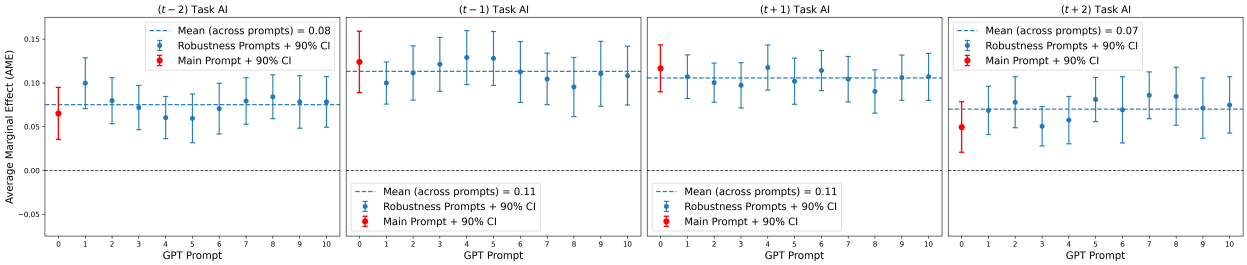
Panel (B): Empirical Fragmentation Index Definition 4



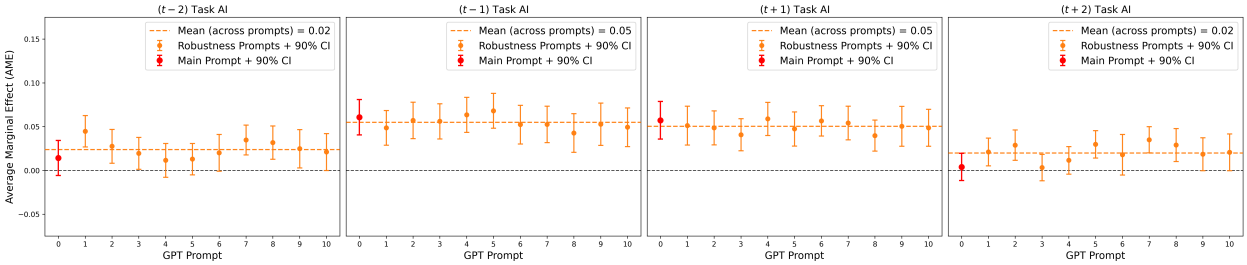
Notes: This figure shows that the estimated relationships between occupation-level AI execution and both AI exposure and the empirical fragmentation index are robust to alternative prompt formulations. Panel (A) reports coefficients on AI exposure (top row, blue) and EFI (bottom row, gold) for execution-based EFI Definition 3. Panel (B) reports the corresponding coefficients for execution-based EFI Definition 4. Error bars denote 90% confidence intervals. The dashed colored lines report the mean across all 11 prompts. In each graph, the Prompt 0 estimate is highlighted in red and corresponds to the estimate reported in Table C.1 using the main prompt. The Panel (A) graphs correspond to columns (1)–(3), and the Panel (B) graphs correspond to columns (4)–(6) of Table C.1.

Figure F.5: Robustness of Table 4 Estimates to GPT Prompts

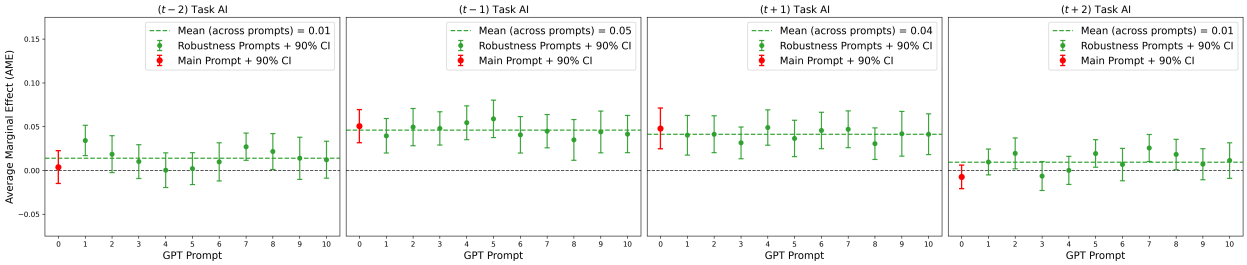
Panel (A): No Fixed Effects



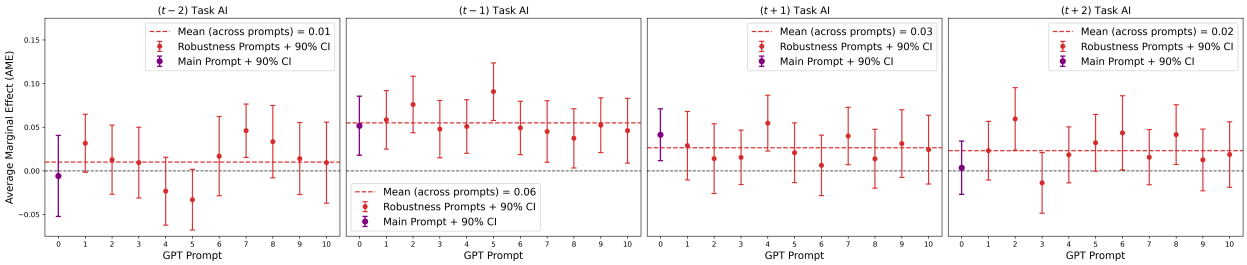
Panel (B): SOC Major Group Fixed Effects



Panel (C): SOC Minor Group Fixed Effects



Panel (D): Detailed Work Activity Fixed Effects



Notes: This figure shows that the spillover effect of neighboring tasks being AI-executed on the focal task's AI execution is robust to alternative prompt formulations. Across prompts, the two immediate neighbors (middle columns) have larger and more precisely estimated effects than more distant neighbors (side columns). The effects of distant neighbors attenuate toward zero and become less distinguishable from zero as additional fixed effects are included. Panels (A)–(D) report the prompt-specific analogs of the estimates in columns (1)–(4) of Table 4. Error bars denote 90% confidence intervals. The dashed colored lines report the mean across all 11 prompts. In each graph, the Prompt 0 estimate corresponds to the estimate reported in Table 4 using the main prompt.