# People Intuitively Schedule Tasks to Improve Collective Efficiency

**Elizabeth Mieczkowski (emiecz@princeton.edu)**[1,*], **Cheryl Li**[2,*], **Natalia Vélez**[3] **& Thomas L. Griffiths**[1,3]
[1]Department of Computer Science, Princeton University
[2]Department of Electrical and Computer Engineering, Princeton University
[3]Department of Psychology, Princeton University
[*]Equal contribution

## Abstract

People routinely collaborate to accomplish goals more quickly, but deciding who should do which tasks, and when, poses a formidable coordination problem. How do people allocate work across collaborators? Related problems have been studied in distributed computer systems, where load balancing algorithms divide heterogeneous tasks among processors with heterogeneous capabilities. Here, we investigate whether human scheduling reflects sophisticated principles from load balancing or simpler heuristics such as taking turns. In a restaurant management task, participants ($N$ = 99) assigned customer orders to chefs with heterogeneous processing speeds, either planning schedules in advance or allocating tasks dynamically. People substantially outperformed simple heuristics, discovering efficient assignments that jointly accounted for collaborator and task heterogeneity. We also uncover systematic biases, including differences between upfront and sequential planning and a preference for completing shorter jobs early. More broadly, linking computational load-balancing theories with cognitive theories of coordination offers a valuable framework for understanding human collaboration.

**Keywords:** collaboration; management; distributed systems; efficiency; load balancing; division of labor

## Introduction

Teamwork is a fundamental part of human cognition. From kitchens and classrooms to emergency rooms and research labs, people routinely collaborate to achieve better outcomes (Pilgrim et al., 2025; Vélez et al., 2023). However, division of labor is an inherently challenging problem. Some collaborators may be better suited to certain tasks than others, and tasks themselves introduce additional complexities, including deadlines, difficulties, and interdependencies. Critically, these decisions must often be made continuously over time as teams assign tasks, complete them, and face new demands.

Despite these challenges, human teams must allocate work rapidly and effectively, whether coordinating who should cook which dishes in a busy restaurant kitchen or dividing tasks during a group project under a tight deadline. A central challenge in these settings is scheduling work so that collaborators can be productively engaged despite differences in task duration, ability, and speed. Given the complexity of this problem, do people rely on simple heuristics such as taking turns, or can they flexibly adapt their allocations to task and team structure?

Closely related problems have been studied extensively in computer science, particularly in the context of *scheduling* and *load balancing*. In distributed computing, load balancing algorithms allocate tasks across multiple servers in order to minimize total processing time (also known as "makespan") (Sharma et al., 2008). Under realistic conditions, where jobs are unpredictable, vary in size and type, and must be processed by heterogeneous servers, optimal allocation is computationally intractable (Braun et al., 2001; Selvakumar & Murthy, 1994). Practical systems must rely on sophisticated algorithms that trade off efficiency, robustness, and computational overhead. These algorithms thus provide a normative benchmark for coordination, specifying how teams *should* allocate tasks over time under different structural constraints. Importantly, this framework is both evaluative and generative: it identifies scenarios in which some strategies are better than others, revealing which features people attend to when their behavior aligns with predictions and diagnosing sources of coordination failures when they diverge.

Here, we test this framework using a restaurant management task ($N$ = 99), in which participants allocate daily sequences of customer orders across chefs with limited capacity and differing skills. We compare human allocations to normative solutions from computer science and simple heuristic strategies. We consider two experimental conditions: an *offline* setting, in which participants must commit to a complete schedule upfront, and an *online* setting, in which participants allocate tasks sequentially as workers complete orders in real time. Although the optimal allocation is identical in both cases, this manipulation allows us to test how different cognitive constraints on planning shape human scheduling, and whether people benefit from real-time observability. We find that people substantially outperform simple heuristics in both regimes, and that performance improves in online settings. By comparing judgments with different load balancing algorithms, we further characterize systematic cognitive biases, such as a preference to prioritize shorter jobs. These results suggest that people can generate task allocations that perform well beyond simple heuristics, systematically approaching normative strategies while accounting for both collaborator and task heterogeneity. Together, our findings provide insight into how people intuitively approach division of labor by drawing on computational theories of load balancing to inform cognitively plausible accounts of human coordination.

## Background

Successful collaboration requires effective task scheduling: dividing sequences of tasks between collaborators. In this



In G. Boleda, I. Dautriche, M. Maldonado, C. Saldana and T. Torrent (Eds.), *Proceedings of the 48th Annual Conference of the Cognitive Science Society* 

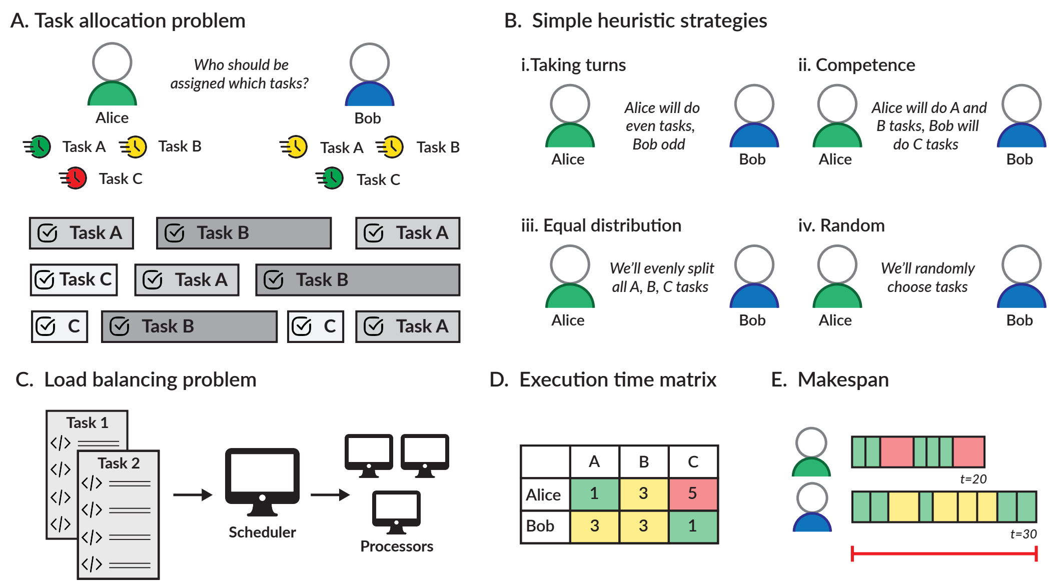


Figure 1: **Task allocation and load balancing. A.** In collaborations, tasks vary in duration and collaborators vary in skill, creating a nontrivial task allocation problem. **B.** To solve this problem, people could adopt a variety of simple strategies, such as (i) taking turns, (ii) assigning each job to the collaborator who is best at it, or (iii) assigning an equal number of each job to each person. **C.** In similar distributed computing problems, a centralized scheduler assigns incoming tasks to a heterogeneous set of processors. **D.** The execution time matrix specifies how long each worker or processor requires to complete each task, capturing heterogeneity in both tasks and collaborators. **E.** The makespan of an allocation (i.e. *total completion time*) is the time when the last worker finishes all of its assigned tasks. More efficient allocations achieve shorter makespans.

section, we summarize existing cognitive science research on this problem, as well as a related literature in distributed computing—specifically, load balancing—which seeks to solve a similar computational problem in groups of machines.

## Human collaboration

Collaboration is fundamental to human progress; people constantly work together to satisfy shared goals, yet these endeavors pose substantial cognitive and organizational challenges (Goldstone & Janssen, 2005; Vélez et al., 2023). Effective collaboration requires solving multiple interdependent coordination problems, including how to divide labor (Grosz & Kraus, 1996), communicate and share information (Hawkins et al., 2021), adapt to the dynamic states of collaborators over time (Wu et al., 2021), and navigate resource and environment constraints (Mieczkowski et al., 2025). These challenges are further compounded by team heterogeneity. Collaborators often differ in roles (Goldstone et al., 2024; Mon-Williams et al., 2025) and skills (Xiang, Vélez, & Gershman, 2023), shaping which divisions of labor are efficient or sustainable. As a result, collaboration requires coordinating temporally interdependent decisions across individuals under cognitive and informational constraints (Griffiths, 2020).

## Task decomposition and scheduling

At the core of these challenges lies the task scheduling problem: deciding who should do what, and when, to achieve collective goals efficiently. Recent work examines how people allocate a set of fixed tasks to collaborators depending on accuracy (Marjieh et al., 2024), finding that people can successfully converge to near-optimal strategies. Related work finds that people account for task, skill, and fairness when collaborating with robots (Chang et al., 2020). However, despite its centrality to collaboration, relatively little is known about the cognitive principles that govern how people schedule sequences of tasks across multiple collaborators.

Related work has largely focused not on collaborative decision-making, but on how individuals strategically decompose and simplify complex tasks in order to reduce combinatorially-exhaustive problems. People can flexibly select heuristics depending on task contexts (Lieder & Griffiths, 2017), plan over long trajectories despite computational constraints (Callaway et al., 2022), and simplify and decompose tasks to mitigate the massive cost of planning (Correa et al., 2023; Ho et al., 2022). Importantly, this work suggests that people can reduce otherwise massive planning problems into feasible and tractable alternatives while maintaining efficiency and accuracy. If these capacities generalize to collaborative contexts, we might expect people to simplify the problem of task scheduling in a way that still preserves efficiency.

## Simple heuristic strategies

Because optimal scheduling is computationally demanding, people may rely on simpler heuristics when deciding who should work on what. One natural strategy is to aim for fairness by splitting work evenly across collaborators. Such strategies may promote more equitable relationships and improved worker well-being (Chen & Saxe, 2024; Scheel et al., 2019), and can be implemented by taking turns or by equally dividing tasks of each type between collaborators. A second class of heuristics reduces computational complexity by ignoring task heterogeneity and focusing only on collaborator competence. Prior work shows that people assign responsibility based on perceived competence (Xiang, Landy, et al., 2023) and use it to reason about how much effort they and their collaborators should exert (Xiang, Vélez, & Gershman, 2023). Similarly, children strategically assign tasks to the most competent collaborator, with a self-serving bias (Baer & Odic, 2022). In our setting, a competence-driven heuristic would involve assigning all tasks of a given type to the collaborator who is best at completing them. Finally, the simplest possible strategy is to allocate tasks randomly. Although this strategy ignores task and team structure, it provides a useful baseline for assessing human behavior. Examples of these heuristic strategies are illustrated in Figure 1B.

### Load balancing algorithms

Although simple heuristics can solve the overall problem of task scheduling, they fail to jointly account for collaborator and task complexity. In distributed computing, load balancing algorithms solve a similar set of problems by making tailored assumptions about task characteristics and available information. These assumptions either introduce additional structure to the problem (Braun et al., 2008; Selvakumar & Murthy, 1994) or sufficiently restrict the solution space to permit efficient computation of the optimal allocation (Johnson et al., 2016; Kafil & Ahmad, 1998). The diversity of the load balancing problem leads to multiple algorithms, each of which result in different assignments (Ali et al., 2005).

The classical heterogeneous load balancing problem involves mapping a set of independent tasks which are known upfront to a set of available processors with fixed execution times. Since both jobs and processors vary, this problem is NP-hard, with no known polynomial-time algorithm for finding optimal assignments (Braun et al., 2001). Similar to the problem of human collaboration, one way that this problem's complexity can be reduced is by using simple heuristics that assign tasks independent of processing time. The round-robin algorithm, for example, allocates tasks in turn-based fashion (Lundberg, 1991). Other simple heuristics allocate tasks by type, ensuring that each processor receives a similar mix of skills or task categories (Chang et al., 2020). More sophisticated algorithms, such as Greedy, Min–Min, and Max–Min, additionally depend on the duration of each task and the cumulative execution time of the sequence of tasks already assigned (Braun et al., 2001). A more detailed description follows in the next section.

## Theoretical Framework

Many real-world collaborations can be viewed as centralized task scheduling problems, in which a decision-maker such as a manager must distribute work across collaborators. Even when all tasks are known in advance, this is a challenging problem; work must be allocated over time to maximize efficiency, avoid idleness, and account for both task and collaborator disparities. Closely related problems have been studied in load balancing. In this section, we formalize task scheduling as a load balancing problem, specifically one with a centralized scheduler in a heterogeneous environment.

We consider a setting with $n = 2$ processors executing a set of $m$ independent tasks $T$ in parallel, with $n < m$. Execution times are specified by an $n \times m$ matrix $E$, where $E_{ij}$ denotes the time required for processor $x_i$ to complete task $t_j$. We assume that both the set of tasks $T$ and the execution times $E$ are known at the time of the assignment. We define a valid allocation as the $n$-tuple $A = (A_1, \ldots, A_n)$, where $A_i$ is the set of tasks assigned to processor $i$, and where the sets $A_1, \ldots, A_n$ are mutually exclusive and collectively exhaustive over $T$. The completion time of an allocation is the time required for all processors to complete their assigned tasks in parallel:

$$\text{completion\_time}(A) = \max_i \Big( \sum_{t_j \in A_i} E_{ij} \Big) \tag{1}$$

The optimal allocation can be computed by enumerating all $n^m$ assignments and selecting the one that minimizes total completion time (Kafil & Ahmad, 1998). Although this exhaustive search guarantees optimality, it is combinatorially expensive and infeasible except for very small problems. Practical systems therefore rely on polynomial-time heuristics that approximate the optimal schedule.

A simple baseline is the Greedy heuristic, which assigns each task to the processor that produces the smallest immediate increase in completion time (Lundberg, 1991). Formally, for any task $t_j$, the assigned processor $x_i$ is chosen by

$$i = \operatorname*{argmin}_i \left( \sum_{t_{j'} \in A_i} E_{ij'} + E_{ij} \right) \tag{2}$$

This strategy yields a locally optimal but globally suboptimal allocation in $O(nm)$ time.

Two widely studied alternatives are the Min–Min and Max–Min heuristics. These algorithms reason not only about the current assignment but also about how heterogeneity in tasks and processors shapes future bottlenecks. For each task, both compute the earliest possible completion time across processors, but differ in which task they prioritize next.

The Min–Min heuristic is "optimistic", assigning first the task $t_j$ with the *smallest* earliest completion time

$$j = \operatorname*{argmin}_j \; \min_i \left( \sum_{t_{j'} \in A_i} E_{ij'} + E_{ij} \right) \tag{3}$$

to the processor that finishes it soonest. Intuitively, Min–Min prioritizes short, easy jobs, rapidly completing small tasks

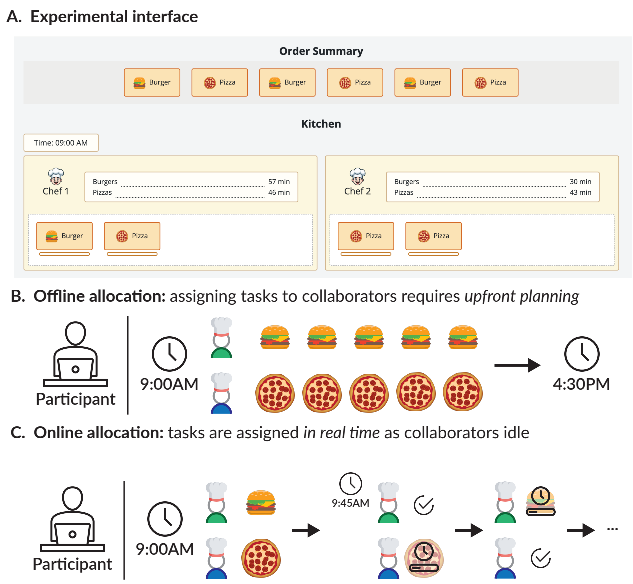


Figure 2: **Restaurant management task. A.** In this experiment, participants act as "managers" of a restaurant, allocating different sets of customer orders (*tasks*) to chefs with different skills. **B.** In the **offline** condition, participants must plan the entire schedule upfront at the beginning of the workday. **C.** In the **online** condition, participants are able to allocate customer orders in real time as chefs become available.

while postponing longer ones. In contrast, the Max–Min heuristic assigns first the task with the *largest* earliest completion time

$$j = \underset{j}{\operatorname{argmax}} \; \min_{i} \left( \sum_{t_{j'} \in A_i} E_{ij'} + E_{ij} \right) \tag{4}$$

again to the processor that completes it most quickly. By prioritizing long jobs, Max–Min prevents difficult tasks from becoming late stragglers that dominate the makespan. Both algorithms run in $O(nm^2)$ time and yield computationally efficient approximations to the optimal schedule.

## Methods

To test whether people rely on simple heuristics or more sophisticated strategies when scheduling tasks, we developed a *restaurant management task* in which participants acted as manager of a restaurant, deciding which of two chefs should complete each customer order. **Task heterogeneity** was operationalized as variability in the types and durations of customer orders, while **collaborator heterogeneity** corresponded to differences in processing times across chefs.

Participants completed simulated workdays by dragging customer orders to one of the available chefs. In the **offline allocation** condition, participants assigned all orders upfront before observing their completion over time. In the **online allocation** condition, participants assigned orders sequentially as chefs completed them. When a chef finished an order, the timer paused and the chef appeared idle until a new order was assigned. In the online interface, participants also had the option to send a chef home rather than assign them a task. Although this behavior would never arise in the optimal allocation, this feature prevented participants from being locked into progressively worse allocations when making sequential decisions. For instance, if a participant had made suboptimal early assignments, forcing them to use a slow available chef rather than waiting for a faster busy chef to finish could compound the error. Figure 2 shows the interface presented to participants during the task.

**Participants.** We recruited 100 English-speaking adults from Prolific in exchange for monetary compensation. One participant failed to complete all trials, resulting in a final sample of 99 participants. No additional exclusion criteria were applied in order to capture a broad range of intuitions about the task. The experiment was approved by the Institutional Review Board, and all participants provided informed consent prior to participation.

**Design and stimuli.** Participants were presented with ten workloads in randomized order that varied along two dimensions: task heterogeneity and collaborator heterogeneity. Each workload was selected to have a uniquely optimal schedule with a standardized makespan of $200 \pm 20$ time units. Guided by an initial large-scale simulation of performance across many combinations of workloads and processors (Figure 3A), we restricted the stimulus set to workloads for which all sophisticated algorithms outperformed all simple heuristics by at least 10 time units, and where either the Min-Min or Max-Min algorithm achieved the optimal allocation.

**Preregistration and key predictions.** This study was preregistered prior to data collection at https://aspredicted.org/w7ic6h.pdf. Specifically, we predicted two key hypotheses: (i) people would significantly outperform any simple heuristic strategy, and (ii) people would find more efficient allocations in the online version of the task compared to offline.

## Results

### Optimal strategies depend on task and collaborator heterogeneity

Do efficient allocations require jointly accounting for both task and collaborator heterogeneity? As an initial test, we used large-scale simulations as a normative baseline to assess how these variables shape which strategies are optimal (Figure 3A). To begin, we simulated a set of 4000 randomly generated workloads, systematically manipulating two dimensions: low versus high task heterogeneity and processor heterogeneity. Across all combinations, sophisticated scheduling algorithms systematically outperformed simple heuristic strategies. However, which sophisticated algorithm performed best

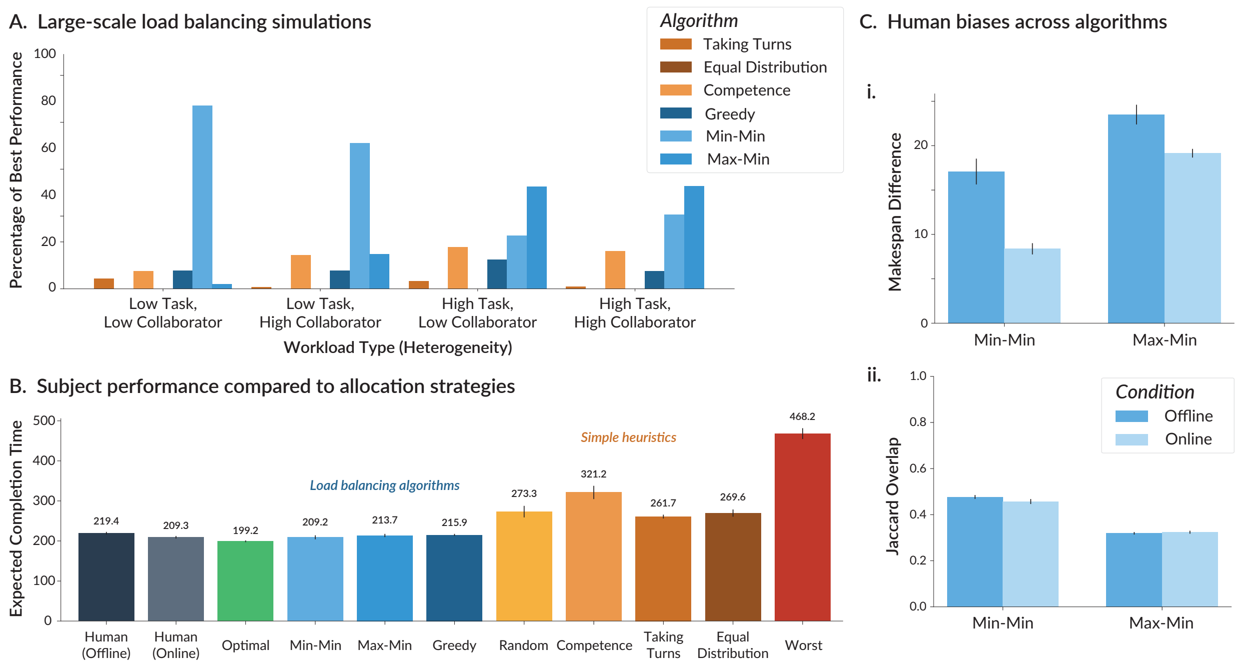


Figure 3: **Results.** **A.** In a set of large-scale load balancing simulations, we randomly generated 4000 workloads that varied in task and collaborator heterogeneity. We compare the makespan of the allocation resulting from each algorithm, and plot the percentage of workloads from each category in which a particular algorithm produced the uniquely optimal allocation that minimized the makespan. **B.** People perform generally well at the restaurant management task, making significantly more efficient allocations than any of the simple heuristics in both the offline and online conditions (error bars: SEM). Participants are more efficient on average in the online condition, when they can utilize real-time information about chef idleness to inform their allocations. "Optimal" corresponds to the single best allocation, or also to the best-performing load balancing algorithm. **C.** Load balancing algorithms reveal that people are biased towards faster progress. Participants are more closely aligned with Min-Min algorithms than Max-Min in terms of both (**i.**) makespan and (**ii.**) allocation similarity (error bars: SEM).

depended on heterogeneity structure. The Min-Min algorithm performed especially well in cases with low task heterogeneity, whereas the Max-Min algorithm more often produced the best allocations when task heterogeneity was high. Intuitively, this finding makes sense. When task durations are relatively heterogeneous, long jobs disproportionately determine the makespan, making it optimal to prioritize these tasks early to prevent late-stage stragglers.

## People allocate tasks efficiently

Participants substantially outperformed all predetermined heuristic strategies, producing more efficient schedules with shorter completion times across workloads (Figure 3B). In both the offline and online conditions, human allocations yielded significantly lower makespans than the best-performing heuristic baseline for a given workload, tested using one-sample t-tests against the heuristic-predicted makespans (offline: $M = -28.02$, $t(49) = -15.95$, $p < .001$, $BF_{10} > 10^{18}$; online: $M = -38.13$, $t(48) = -47.06$, $p < .001$, $BF_{10} > 10^{39}$). This advantage suggests that participants were able to jointly integrate information about workloads and collaborators in ways that exceeded the fixed assumptions encoded in the heuristic baselines.

## Performance improves with real-time observability

In both conditions, participants converged on efficient solutions that outperformed all heuristic strategies. However, they performed significantly better in the online condition, using real-time feedback about worker idleness to update their strategies within trials (Figure 3B; compare Human (Offline) vs. Human (Online)). We fit a linear mixed-effects model predicting the human-heuristic makespan difference from condition (offline vs. online), with random intercepts for participants and workloads. Participants exhibited a significant advantage in online scheduling compared to offline ($\beta = -10.11$, $z = -58.85$, $p < .001$), indicating more efficient task allocations when participants could update their decisions based on ongoing feedback.

## People preferentially prioritize shorter jobs

Comparing human allocations with the predictions of load balancing algorithms provides a unique opportunity to test and interpret potential biases in task scheduling. In two exploratory analyses, we examined whether human schedules more closely resembled the Min-Min algorithm, which prioritizes completing short jobs first, or the Max-Min algorithm, which prioritizes longer jobs to prevent later bottlenecks. First, we analyzed differences in makespan (Figure 3C(i)). Human allocations were significantly closer to Min-Min than to Max-Min in both the offline condition ($M = -6.42$, $t(49) = -6.35$, $p < 0.001$, $BF_{10} > 10^5$) and the online condition ($M = -10.77$, $t(48) = -13.55$, $p < 0.001$, $BF_{10} > 10^{15}$). Second, we examined fine-grained similarities between human and algorithmic assignments using the Jaccard index, which measures how many task-worker assignments are shared between two schedules relative to the total number of distinct assignments across both (Figure 3C(ii)). The sets of tasks assigned by participants to each worker were significantly more similar to Min-Min than to Max-Min in both the offline condition ($M = 0.16$, $t(49) = 14.80$, $p < 0.001$, $BF_{10} > 10^{16}$) and the online condition ($M = 0.13$, $t(48) = 9.79$, $p < 0.001$, $BF_{10} > 10^{10}$). Together, these analyses indicate that participants systematically allocated tasks in a manner consistent with Min-Min reasoning, favoring early progress and shorter jobs, both in terms of performance outcomes and assignment structure.

### Statistical power

Post-hoc power analysis confirmed that all tests were substantially overpowered: observed effect sizes ranged from Cohen's $d = 0.91$ to 6.66, and achieved power was 1.00 for every primary and exploratory one-sample t-test. Bayesian analysis using JZS priors (Rouder et al., 2009) corroborated these results: Bayes factors for all comparisons were decisive ($BF_{10} > 10^5$ in every case, reaching $BF_{10} > 10^{39}$ for the primary hypothesis in the online condition), confirming that the evidence is robust independent of inferential framework. Posterior distributions of Cohen's $d$ under a flat prior showed 95% credible intervals entirely excluding zero for all effects.

## Discussion

Task scheduling is a fundamental challenge faced by collaborative teams, and a daunting one. Theories from computer science, specifically distributed computing, show that heterogeneity in tasks and collaborators introduces a combinatorially complex problem. Despite this difficulty, human teams must routinely divide labor to accomplish shared goals. Do people solve this challenge by relying on simple heuristics like taking turns to avoid cognitively costly planning, or can they generate more sophisticated assignments that jointly account for task and team structure?

In this work, we introduce a theoretical framework from distributed systems—load balancing—that provides a normative baseline for which strategies should be optimal under different forms of heterogeneity. In a preregistered behavioral experiment, we find that people systematically produce sophisticated task allocations that resemble these algorithms and substantially outperform simple heuristics. We further show that people benefit from real-time observability, which allows them to take into account the dynamic states of collaborators. Finally, this framework provides an interpretable lens for understanding systematic biases in human scheduling, including a preference for completing shorter jobs early even when longer ones ultimately create bottlenecks.

This work presents a preliminary bridge between human collaboration and distributed computing, opening a generative space for future research in cognitive science. Here, we focus on a specific class of problems in which the workload is known upfront, tasks and processors are heterogeneous, and a single centralized scheduler assigns tasks. Each of these assumptions can be relaxed, yielding a range of theoretically and empirically rich extensions. For example, what if the system is "decentralized" with no manager, such that collaborators must claim and complete tasks simultaneously? Centralized systems with a single scheduler benefit from global coordination, but suffer from bottlenecks such as waiting for task assignments. Decentralized algorithms avoid these delays at the cost of communication and consensus overhead (Mieczkowski et al., 2026). Understanding how humans navigate these trade-offs represents a promising direction for future work. Another assumption that can be relaxed is the static workload. Dynamic load balancing algorithms provide normative predictions for how tasks should be allocated in real time if workloads are unknown or fluctuate. Studying how humans approximate or deviate from these strategies offers a natural extension of this framework. Similarly, what if workers themselves are unknown or stochastic? In many real-world settings, workers' processing rates must be inferred from experience, and are noisy depending on fatigue, task-switching, and failure. Ideas such as queueing theory, which accounts for stochastic behavior of individual servers, could be extended to predict outcomes under these constraints.

Throughout this work we have emphasized cases in which people successfully approximate efficient scheduling strategies. However, coordination is not always effortless, and real-world collaborations frequently break down. Connecting cognitive models of collaboration with normative predictions from distributed computing helps explain not only when coordination succeeds, but also *where* and *why* it may fail. This framework offers a foundation not only for understanding collaborative success, but also for designing tools, interfaces, and organizational structures that help teams anticipate, detect, and correct coordination failures when they arise. Finally, a natural step toward understanding the cognitive foundations of collaboration is to move beyond comparing behavioral outputs to algorithmic benchmarks, toward identifying the underlying processes people actually use and characterizing naturalistic "human" scheduling algorithms.

### Code Availability

The stimuli, analysis code, and data used in this study are publicly available at https://github.com/emieczkowski/TaskScheduling_for_HumanCollab.

## Acknowledgments

This work was supported by the National Defense Science and Engineering Graduate (NDSEG) Fellowship Program to EM, and ONR MURI N00014-24-1-2748 to TG.

## References

Ali, S., Braun, T. D., Siegel, H. J., Maciejewski, A. A., Beck, N., Bölöni, L., Maheswaran, M., Reuther, A. I., Robertson, J. P., Theys, M. D., & Yao, B. (2005). Characterizing resource allocation heuristics for heterogeneous computing systems. In *Advances in Computers* (pp. 91–128, Vol. 63). Elsevier.

Baer, C., & Odic, D. (2022). Mini managers: Children strategically divide cognitive labor among collaborators, but with a self-serving bias. *Child Development*, *93*(2), 437–450.

Braun, T. D., Siegel, H. J., Beck, N., Bölöni, L. L., Maheswaran, M., Reuther, A. I., Robertson, J. P., Theys, M. D., Yao, B., Hensgen, D., & Freund, R. F. (2001). A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing*, *61*(6), 810–837.

Braun, T. D., Siegel, H. J., Maciejewski, A. A., & Hong, Y. (2008). Static resource allocation for heterogeneous computing environments with tasks having dependencies, priorities, deadlines, and multiple versions. *Journal of Parallel and Distributed Computing*, *68*(11), 1504–1516.

Callaway, F., van Opheusden, B., Gul, S., Das, P., Krueger, P. M., Griffiths, T. L., & Lieder, F. (2022). Rational use of cognitive resources in human planning. *Nature Human Behaviour*, *6*(8), 1112–1125.

Chang, M. L., Pope, Z., Short, E. S., & Thomaz, A. L. (2020). Defining fairness in human-robot teams. *29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 1251–1258.

Chen, A. M., & Saxe, R. (2024). How taking turns communicates desired equality in social relationships. *Proceedings of the Annual Meeting of the Cognitive Science Society*, *46*.

Correa, C. G., Ho, M. K., Callaway, F., Daw, N. D., & Griffiths, T. L. (2023). Humans decompose tasks by trading off utility and computational cost. *PLoS Computational Biology*, *19*(6), e1011087.

Goldstone, R. L., Andrade-Lotero, E. J., Hawkins, R. D., & Roberts, M. E. (2024). The emergence of specialized roles within groups. *Topics in Cognitive Science*, *16*(2), 257–281.

Goldstone, R. L., & Janssen, M. A. (2005). Computational models of collective behavior. *Trends in Cognitive Sciences*, *9*(9), 424–430.

Griffiths, T. L. (2020). Understanding human intelligence through human limitations. *Trends in Cognitive Sciences*, *24*(11), 873–883.

Grosz, B. J., & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence*, *86*(2), 269–357.

Hawkins, R. D., Gweon, H., & Goodman, N. D. (2021). The division of labor in communication: Speakers help listeners account for asymmetries in visual perspective. *Cognitive Science*, *45*(3), e12926.

Ho, M. K., Abel, D., Correa, C. G., Littman, M. L., Cohen, J. D., & Griffiths, T. L. (2022). People construct simplified mental representations to plan. *Nature*, *606*(7912), 129–136.

Johnson, L. B., Choi, H.-L., & How, J. P. (2016). The role of information assumptions in decentralized task allocation: A tutorial. *IEEE Control Systems Magazine*, *36*(4), 45–58.

Kafil, M., & Ahmad, I. (1998). Optimal task assignment in heterogeneous distributed computing systems. *IEEE Concurrency*, *6*(3), 42–50.

Lieder, F., & Griffiths, T. L. (2017). Strategy selection as rational metareasoning. *Psychological Review*, *124*(6), 762.

Lundberg, L. (1991). Static process allocation using information about program behaviour. *Proceedings of the Twenty-Fourth Annual Hawaii International Conference on System Sciences*, *1*, 1–9.

Marjieh, R., Gokhale, A., Bullo, F., & Griffiths, L. (2024). Task allocation in teams as a multi-armed bandit. *ACM Collective Intelligence*.

Mieczkowski, E., Collins, K. M., Sucholutsky, I., Vélez, N., & Griffiths, T. L. (2026). Language model teams as distributed systems. *arXiv preprint arXiv:2603.12229*.

Mieczkowski, E., Turner, C., Vélez, N., & Griffiths, T. L. (2025). People evaluate idle collaborators based on their impact on task efficiency. *Cognition*, *264*, 106200.

Mon-Williams, R., Taylor-Davies, M., Mieczkowski, E., Velez, N., Bramley, N. R., Wang, Y., Griffiths, T. L., & Lucas, C. G. (2025). Partner modelling emerges in recurrent agents (but only when it matters). *arXiv preprint arXiv:2505.17323*.

Pilgrim, C., Morford, J., Warren, E., Aellen, M., Krupenye, C., Mann, R. P., & Biro, D. (2025). The computational foundations of collective intelligence. *arXiv preprint arXiv:2509.07999*.

Rouder, J. N., Speckman, P. L., Sun, D., Morey, R. D., & Iverson, G. (2009). Bayesian t tests for accepting and rejecting the null hypothesis. *Psychonomic bulletin & review*, *16*(2), 225–237.

Scheel, T. E., Otto, K., Vahle-Hinz, T., Holstad, T., & Rigotti, T. (2019). A fair share of work: Is fairness of task distribution a mediator between transformational leadership and follower emotional exhaustion? *Frontiers in Psychology*, *10*, 2690.

Selvakumar, S., & Murthy, C. S. R. (1994). Static task allocation of concurrent programs for distributed computing systems with processor and resource heterogeneity. *Parallel Computing*, *20*(6), 835–851.

Sharma, S., Singh, S., & Sharma, M. (2008). Performance analysis of load balancing algorithms. *World Academy of Science, Engineering and Technology*, *38*(3), 269–272.

Vélez, N., Christian, B., Hardy, M., Thompson, B. D., & Griffiths, T. L. (2023). How do humans overcome individual computational limitations by working together? *Cognitive Science*, *47*(1), e13232.

Wu, S. A., Wang, R. E., Evans, J. A., Tenenbaum, J. B., Parkes, D. C., & Kleiman-Weiner, M. (2021). Too many cooks: Bayesian inference for coordinating multi-agent collaboration. *Topics in Cognitive Science*, *13*(2), 414–432.

Xiang, Y., Landy, J., Cushman, F. A., Vélez, N., & Gershman, S. J. (2023). Actual and counterfactual effort contribute to responsibility attributions in collaborative tasks. *Cognition*, *241*, 105609.

Xiang, Y., Vélez, N., & Gershman, S. J. (2023). Collaborative decision making is grounded in representations of other people's competence and effort. *Journal of Experimental Psychology: General*, *152*(6), 1565.