return results (e.g., MapReduce [34]).

**Managing producer/consumer and task/subtask relationships**

In many situations, one activity produces something required as input for another activity. For example, the structure of an article needs to be decided on before the sections can be written. These same requirements exist in distributed computing, in which tasks need to be scheduled so that they can be completed in the correct sequence and in a timely manner, with data being transferred between computing elements appropriately. Deciding how to divide a task into subtasks and managing those subtasks is also a challenging problem, especially for complex and interdependent tasks [61,89]. This is true whether a manager in an organization is trying to plan a large project or a programmer is trying to parallelize a complex task. Furthermore, top-down approaches in which a single person (e.g., the task creator) specifies all subtasks a priori may not be possible, or subtasks may change as the task evolves.

**Crowd-Specific Factors**

Unlike traditional organizations in which workers possess job security and managers can closely supervise and appropriately reward or sanction workers, or distributed computing systems in which processors are usually highly reliable, crowd work poses unique challenges for both workers and requesters ranging from job satisfaction to direction-setting, coordination, and quality control. For example, organizations can maintain high quality work through management, worker incentives, and sanctions.

While some of these methods are available in crowd work (e.g., how much to reward workers, whether to reject their work, or impose a reputation penalty) their power is attenuated due to factors such as lack of direct supervision and visibility into their work behavior, lack of nuanced and individualized rewards, and the difficulty of imposing stringent and lasting sanctions (since workers can leave with fewer repercussions than in traditional organizations, such as to reference letters or work histories). The worker’s power is also limited: requesters do not make a long-term commitment to the worker, and endure few penalties if they renge on their agreement to pay for quality work. In distributed computing systems, by contrast, requesters (programmers) have fewer problems with motivating and directing their workers (computers). However, machines cannot match the complexity, creativity, and flexibility that human intelligence manifests. Combining ideas from human and computer organization theories may thus provide complementary benefits and address complementary weaknesses over using either alone.

**Framework**

Figure 2 presents a framework that integrates the challenges posed by managing shared resources (such as assigning workers to appropriate tasks), managing producer-consumer relationships (such as decomposing tasks and assembling them into a workflow), and crowd-specific factors (such as motivation, rewards, and quality assurance). Many of its elements combine insights from organizational behavior and distributed computing: for example the task decomposition and task assignment functions use both human and computational processes.

The goal of this framework is to envision a future of crowd work that can support more complex, creative, and highly valued work. At the highest level, a platform is needed for managing pools of tasks and workers. Complex tasks must be decomposed into smaller subtasks, each designed with particular needs and characteristics which must be assigned to appropriate groups of workers who themselves must be properly motivated, selected (e.g., through reputation), and organized (e.g., through hierarchy). Tasks may be structured through multi-stage workflows in which workers may collaborate either synchronously or asynchronously. As part of this, AI may guide (and be guided by) crowd workers. Finally, quality assurance is needed to ensure each worker’s output is of high quality and fits together.

Because we are concerned with issues of design – the technical and organizational mechanisms surrounding crowd work – we highlight in the process model twelve specific research foci (Figure 2) that we suggest are necessary for realizing such a future of crowd work. These foci are grouped into three key dimensions: foci relevant to the work process; the computation guiding, guided by, and underlying the work; and the workers themselves. Our 12 foci overlap each other in places. However, in total they provide a wide-ranging multidisciplinary view that covers

![Figure 2: Proposed framework for future crowd work processes to support complex and interdependent work.](image-url)