How we write with crowds

MOLLY Q FELDMAN, Cornell University, USA
BRIAN MCINNIS, University of California San Diego, USA

Writing is a common task for crowdsourcing researchers exploring complex and creative work. To better understand how we write with crowds, we conducted both a literature review of crowd-writing systems and structured interviews with designers of such systems. We argue that the cognitive process theory of writing described by Flower and Hayes [16], originally proposed as a theory of how solo writers write, offers a useful analytic lens for examining the design of crowd-writing systems. This lens enabled us to identify system design challenges that are inherent to the process of writing as well as design challenges that are introduced by crowdsourcing. The findings present both similarities and differences between how solo writers write versus how we write with crowds. To conclude, we discuss how the research community might apply and transcend the cognitive process model to identify opportunities for future research in crowd-writing systems.

CCS Concepts: • **Human-centered computing → Collaborative and social computing systems and tools**; *HCI theory, concepts and models.*

Additional Key Words and Phrases: crowd-writing; crowdsourcing; writing; writing theory; collaborative writing

ACM Reference Format:
Molly Q Feldman and Brian McInnis. 2020. How we write with crowds. *Proc. ACM Hum.-Comput. Interact.* 4, CSCW2, Article *Preprint* (2020), 31 pages.

1 INTRODUCTION

Writing is a powerful form of communication which most people encounter daily. Despite its pervasiveness, learning how to write well is not easy [6]. One successful method writers use is to decompose the act of writing into a process that involves a collection of tasks performed by the writer(s). Research about how people write has appeared in a wide range of academic disciplines, from psychology to education to computer science. For example, Flower and Hayes [16] detail a series of steps involved with writing as a solo writer.

While being a solo writer can be difficult, writing with multiple authors introduces additional complexity [40]. Effectively working with multiple authors requires understanding how people write together and how they communicate [3, 35, 45]. Wikipedia is one example of performing collaborative writing online and at-scale between writers working independently [71]. Yet, even in distributed collaborative efforts, like Wikipedia, writers have some control over what topics they choose to write about and how they choose to participate. Additionally, Wikipedia contributors have some understanding of how their work contributes to the collaborative effort.

Over the last decade a new form of crowdsourced writing has emerged in which online, distributed groups of workers with minimal mutual interaction [59] are brought together to generate

Authors' addresses: Molly Q Feldman, mqf3@cornell.edu, Cornell University, USA; Brian McInnis, bmcinnis@ucsd.edu, University of California San Diego, USA.

2573-0142/2020/0-ART*Preprint* $15.00
https://doi.org/

content and assemble it into writing. Although some crowdsourcing systems have produced writing of similar quality to that produced by a solo writer, it has been challenging to achieve such success. In order to crowdsource writing, system designers have had to overcome many socio-technical challenges related to task design and decision-making across multiple layers of text-heavy information.

In this paper, we reflect on the development of such systems for generating original writing with crowds (hereafter referred to as *crowd-writing systems*), by examining the following research question:

What are the design decisions behind existing crowd-writing systems?

The paper presents a structured overview of crowd-writing systems research, which may serve as a reference guide for those looking to build new crowd-writing systems and as an opportunity for the research community involved in this work to consider its past, current, and future system design practices. The analysis involved a literature review of CHI, UIST, CSCW, and HCOMP[1] proceedings as well as interviews with many of the paper authors, so as to better understand the development story and design practices behind each of the crowd-writing systems. The findings are based on an analysis of 24 crowd-writing systems papers.

Through our analysis, we found it useful to apply the cognitive process theory of writing developed by Linda Flower and John R. Hayes [16] as an analytic lens for studying the design of crowd-writing systems. This lens enabled us to identify design challenges that are inherent in the process of writing as well as design challenges that are introduced by crowdsourcing. The findings present both similarities and differences between how solo writers write versus how we write with crowds. Specifically, we find that crowd-writing systems are built embodying a set of steps that are similar to solo writing. However, crowd-writing systems also introduce new writing challenges because the systems may involve many workers in the writing process, often contributing independently. These challenges include how to assign workers to writing tasks, how to integrate task work into sentences and paragraphs in drafts of the writing, and how to maintain coherence and monitor progress throughout the writing process.

Unlike a solo writer, writing with the crowd also requires significant system design attention to the worker as a person as well as how workers engage with each other and the writing tasks collectively. This analysis reaffirms the importance of considering workers holistically when developing crowd-powered systems for complex domains [27, 49]. Finally, we suggest opportunities for future work concerning the practical task of building crowd-writing systems and open questions regarding specific writing processes.

2 BACKGROUND

2.1 Why study writing with crowds?

Writing tasks are a frequent context for research about what complex and creative work crowdsourcing systems might be able to accomplish. One definition of crowdsourcing is "taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call" [24]. The open call tends to be via an online labor market (see Table 1), which enables "requesters to post tasks to workers to perform in return for monetary payment" [39, pg. 46]. Crowdsourcing tasks are typically small, short pieces of work called micro-tasks [13].

[1]These are abbreviations for the following conferences: ACM CHI Conference on Human Factors in Computing Systems (CHI), ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW), ACM User Interface Software and Technology Symposium (UIST), and AAAI Conference on Human Computation and Crowdsourcing (HCOMP).

The distributed nature of crowds means that people must be coordinated to collectively accomplish tasks. Coordination, defined as managing dependencies among activities [48], can take multiple forms: passing information from one person to another or monitoring the behavior of others. Coordination theory indicates that systems typically introduce fixed roles and hierarchies about how information and work is allocated to different members [48]. Considerations include resource allocation, producer-consumer relationships between group members, and interdependencies between different parts of the system.

Accomplishing work in a crowdsourcing system requires workers to have some level of skill. Traditionally, micro-tasks have been designed so that they draw from a person's basic skills, rather than their specialized knowledge [32]. As crowdsourcing tasks require different types of skill, some crowdsourcing systems may provide on-the-job training as workers perform micro-tasks [2]. On the other hand, some crowdsourcing systems recruit highly-skilled professionals who can be trusted to perform each task correctly and with minimal oversight [56].

Although crowdsourcing research has explored a wide range of skilled application domains (e.g., accessibility [9], scheduling [12], online curriculum development [56]), this paper focuses on the application domain of writing for several reasons. First, editing text has been called the *white lab rat* of HCI research [64], because writing is a prototypical task for examining human-computer interactions. With respect to crowdsourcing systems research, iterative writing tasks were among the first explored with the human-computation protocol *TurKit* [44]. A notable takeaway from research with TurKit is that crowd workers can effectively write and improve paragraphs.

As a second source of motivation, writing has unique properties as an application domain, especially when compared to other crowdsourcing applications. Unlike traditional micro-task domains, where the actual mechanics of the micro-task may be clear (e.g., clicking a button), it is not obvious how to best decompose a writing task into micro-tasks and workflows. At the same time, writing exhibits explicit, repeatable structure [16] not found in other complex domains for crowdsourcing (e.g., art [55], interface design [37]). The combination of an interpretable task that is difficult to decompose makes crowd-writing an interesting area of research.

Although writing is important to the HCI research community and a unique application area, systems which write with crowds have not been considered a distinct area of crowdsourcing research, prior to this paper. One potential reason is that crowdsourcing systems have integrated tasks related to writing for a variety of purposes. Some systems [1, 21, 31] generate complete stories or articles, whereas other systems incorporate writing tasks in the process of accomplishing objectives beyond writing [72, 73, 75]. The intent for some crowd-writing systems is to exhibit the exciting potential of crowdsourcing (e.g., [33]), but this is not the case for all systems, as the purpose for many is simply to assist in the writing of something (e.g., [7]).

2.2 How to study writing with crowds?

There are two practical challenges involved with a structured analysis of crowd-writing system design: (1) How might we compare different writing tasks, and (2) How might we compare the socio-technical environments for writing associated with each system. For the purpose of this research, we chose to compare writing tasks by abstracting away the mechanics of writing in favor of a model of the writing process [16]. To account for differences among the socio-technical environments for crowd-writing, we draw upon concepts from existing research about collaborative writing [3, 35, 45].

2.2.1 The cognitive process model for writing. Cognitive models of writing [6, 41] provide a comprehensive starting point for analyzing the writing process of any writer. Flower and Hayes [16] present a well-known cognitive model of writing: *A Cognitive Process Theory of Writing*. In this

section, we provide an overview of this specific model of writing [16], which we will refer to throughout this paper as "the cognitive process model."

The cognitive process model presents a solo writer's writing process as a series of **components** (Figure 1). Each component represents a step of writing where specific tasks occur or decisions are made. The overall model organizes these components in a recursive, interconnected, and hierarchical way; solo writers can go back and forth, up and down, within the model from component to component. The structure is different from previous staged models, which view writing as a linear process, moving from one stage to the next [6].

The components of the model are organized into three major units: **Task Environment**, **Writing Processes**, and **Long-Term-Memory**. At a high level, the Task Environment and the Long-Term Memory interface with the Writing Processes and provide the context in which the solo writer writes. The Task Environment contains everything external to the solo writer themselves, most notably the writing assignment, set by a teacher or another person. The Long-Term Memory contains the solo writer's own knowledge of different topics (e.g., the American government structure, types of wheat) and different audiences (e.g., doctors, library patrons). The solo writer then uses the context provided by the Task Environment and the Long-Term Memory to inform the Writing Processes.

The Writing Processes describe how the solo writer generates the final text. The Planning component is where the solo writer gathers information from the Task Environment and their Long-Term Memory to create a writing plan. The solo writer first generates ideas in the Generating step, organizes them in Organizing, and then sets explicit goals in the Goal-setting step. During Translating, the solo writer uses their goals to generate sentences and paragraphs which become the Text Produced So Far. The solo writer reads, edits, and revises the Text Produced So Far in Reviewing. Throughout, a solo writer uses their Monitor process to manage transitions between components and to coordinate among the components.

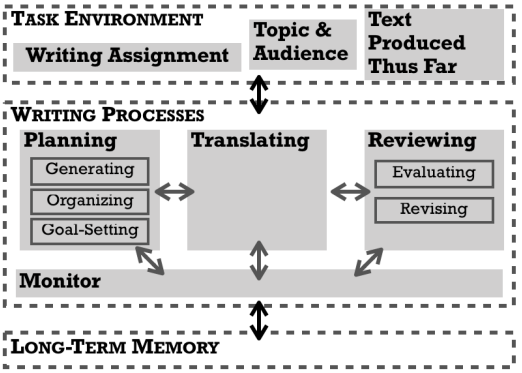

Fig. 1. A diagram of Flower & Hayes's a cognitive process theory of writing [16]. Dotted boxes represent the three major units, solid boxes represent components within those units, and solid line boxes represent subcomponents.

2.2.2 Collaborative writing. While the focus of this paper is not on solo writers, but rather on systems that help distributed crowd workers to write together, the cognitive process model remains relevant. Regardless of how multiple writers are organized, more than one writer increases the complexity of the writing process. The situation where multiple writers explicitly work together in a well-defined group or team typically falls under the category of **collaborative writing**. Collaborative writing is defined as "an iterative and social process that involves a team focused on a common objective that negotiates, coordinates, and communicates during the creation of a common document" [45, pg. 72].

Collaborative writing is a complex process, yet it is somewhat similar to the cognitive process model of writing for solo writers. In a study on communication modalities for student group collaboration, Kraut et al. [35] found that all groups, regardless of modality, exhibit basic writing

flows similar to a solo writer. Specifically, Kraut et al. [35] find that the cognitive process model presented in Flower and Hayes [16] is consistent with the process of collaborative writing:

> "As one would expect, on average, groups plan before they write and write before they revise; the activities peak in this order. However, the phase structure of collaborative projects is very loose, and these activities overlap. Early in the project, the typical group was planning and drafting concurrently, and, later in the project, the typical group was drafting and revising concurrently. This finding is consistent with descriptions of individual writing processes [16]" [35, pg. 386].

Writing theory research has cataloged numerous strategies that groups have used to collaboratively write [3, 35, 45]. Some groups appoint a single writer to draft a document that the group then revises (single-writer writing) whereas other groups all contribute to drafting, but do so either at the same time (parallel writing) or one after another (sequential writing), whether in close proximity or not [45]. Additionally, group members exhibit different levels of influence or agency over the writing and group decision-making, e.g., collaborative writing groups where there is an accepted leader [3]. Regardless of the overall style, researchers find Planning a document to typically be a group task whereas Translating tends to be an individual task [3].

2.3 Research Question

Writing alone [16], collaborative writing [3, 35, 45], and crowd-writing offer different approaches to the process of writing. The purpose of this paper is to examine crowd-writing system design as an area of research that is different from writing alone and writing collaboratively. We specifically address the following research question:

What are the design decisions behind existing crowd-writing systems?

For several reasons, we chose to apply the cognitive process model as an analytic lens with which to explore this research question. First, the model describes a solo writer's process, though Kraut et al. [35] found that the cognitive process model is also consistent with collaborative writing processes. Second, in a review of numerous cognitive processes for writing, Becker [6] notes that more recent models consistently use the general outline of this cognitive model as a template. Third, a few crowd-writing systems have drawn direct inspiration from this model (e.g., [29, 30]). These points speak to the model's robustness, generalizability, and applicability to crowd-writing.

3 METHODS

To investigate our research question, we conducted a systematic literature review of crowd-writing systems as well as an interview-based study with crowd-writing system designers. Through the analysis we found it useful to apply the cognitive process model as an analytic lens to examine each system component.

3.1 Definitions

Many terms are used inconsistently within the crowd-writing literature due to reasonable differences in context and author preferences. To avoid confusion, the remainder of this paper will use the following terminology and associated definitions:

Definitions about systems.

- **Crowd-writing system:** a crowd-powered system that produces original writing, as part of an intermediary writing process (e.g., Editor communication [57]) or as a final output of the system (e.g., Mechanical Novel [31]).

- **Written output:** the writing produced by a full run of a writing system, be it a solo writer, collaborative group, or crowd-writing system. Depending on the design, the written output can range from a draft of a paper [52] to a news article [1]. By contrast, the Text Produced So Far is characterized in the cognitive process model as an intermediary output [16].

Definitions about people.[2]

- **Solo writer:** restricted to the setting of one person writing alone, it refers to the writer role defined in Flower & Hayes [16], or the person engaged in various steps of the writing cognitive process model.
- **Worker:** a person who performs tasks associated with a crowd-writing system.
- **Designer:** the person or team behind the creation of a crowd-writing system. The designer is in charge of every aspect of adapting a writing task to the crowd environment, including instructions, recruitment, etc.
- **Client:** the individual or group of individuals that receive the output of the writing task, typically the people intended as the recipients of the writing effort. For a system with real world deployment [1, 7, 47, 52] the client is some target individual or individuals, but often, particularly for many of the research prototypes considered here, the client may simply be the designer.

3.2 Literature search

The literature review includes articles published in four main venues for crowdsourcing research: CHI, CSCW, UIST, and HCOMP. In February 2019, we searched the ACM and AAAI digital libraries for all papers containing the keywords "crowdsourcing" and "writing". The search returned 328 papers published between 2009 and 2018. Both authors of this article read the title and abstract of each paper returned by the search. A paper was excluded from further consideration if its main focus was not crowdsourcing or if the reported work did not involve the generation of any written output or byproduct (e.g., [10, 62]). This resulted in a list of 59 papers that received further consideration.

Each of the remaining 59 papers were read by both authors of this article. This fine-grain analysis led to the removal of papers that either (1) did not meet the definition of a crowd-writing system (Section 3.1), such as systems for real-time captioning [36] that do not produce original writing, or (2) were one of several papers from a group reflecting a body of work from which only a single paper was selected (e.g., Chorus [38] was selected and therefore Evorus [26] was excluded). This final selection process resulted in the set of 24 papers presented in Table 1, together with an overview of their specific approaches to crowd-writing.

Next, we conducted analytic memo writing [61] to explore how each paper's research contributions may relate to components of the cognitive process model [16]. First, we gathered design decisions regarding the system, its evaluation, and any further implications. Each author then independently sorted the design decisions into one of the components. Afterwards, we merged those results and came to consensus about whether each design decision related to the cognitive process model or not and, if so, to which component. Through consensus we iteratively developed a set of design decisions associated with each crowd-writing system, then applied the cognitive process model to examine all design decisions within each writing component.

[2]Note that the crowdsourcing literature often uses the term "Requester" to refer to people that, in our setting, could be either designers or clients. Further, a Requester might take on multiple roles in the system and also participate in the overall crowd-writing effort. Since these distinctions are important to our analysis, we avoid using the term "Requester" and the resulting potential ambiguity or confusion it might create.

	Name	Venue	Written Output	Crowd
[29]	*N/A*	CSCW 18	Action plans for academic papers & Wikipedia	AMT, etc.
[42]	CrowdIA	CSCW 18	Narrative explanation and analysis of an intelligence scenario	AMT
[50]	*N/A*	CSCW 18	Discussion prompts about the MTurk participation agreement	AMT
[68]	CrowdSCIM	CSCW 18	Summaries of historical documents	AMT
[31]	Mechanical Novel	CSCW 17	Original & revised drafts of short stories	AMT
[57]	*N/A*	CSCW 17	Edits to 66 initial mediocre paragraphs	Numerous
[74]	Wikum	CSCW 17	Summary trees of existing discussions on social, political & science topics	Students
[18]	Dear Beta/Gamma	CSCW 16	Hints for resolving verification errors for electronic circuits	Students
[30]	Storia	CSCW 16	Four-sentence paragraphs summarizing world events	AMT
[46]	CrowdCrit	CSCW 15	Structured feedback on posters for a design contest	AMT
[73]	Voyant*	CSCW 15	Structured and free-form feedback on event posters for a course project	AMT
[75]	*N/A*	CSCW 14	Numerous (reviews, ideas, summaries)	AMT
[14]	*N/A*	CSCW 12	Reviews for 6 consumer products	AMT
[47]	CommunityCrit	CHI 18	Recommendations for urban design	Community
[21]	Knowledge Accelerator	CHI 16	Articles answering open-ended questions	AMT
[52]	WearWrite	CHI 16	First drafts of writing projects (academic paper introduction or blog post)	AMT
[72]	*N/A*	CHI 12	Explanations of data visualization trends	AMT
[38]	Chorus	UIST 13	Answers to user-generated questions (e.g., tourist information, recipe ideas)	Numerous
[33]	CrowdForge	UIST 11	Popular news summary of an academic journal article	AMT
[7]	Soylent	UIST 10	New drafts of existing texts	AMT
[15]	MicroTalk	HCOMP 16	Justifying explanations for relation extraction tasks	AMT
[1]	*N/A*	HCOMP 15	Listicle articles chronicling local events	Numerous
[43]	*N/A*	HCOMP 14	Explanations of internet memes	AMT
[66]	Context Trees	HCOMP 14	Summaries of large pieces of content (e.g., a short story, a movie)	AMT

Table 1. The 24 papers that make up our literature review. **Name** is the name given to the system described by the paper, *N/A* if none provided; **Venue** is the conference name and publication year; **Written Output** summarizes what type of written content the system produces; **Crowd** is the worker recruitment mechanism (e.g., Amazon's Mechanical Turk (AMT) [4]). *Numerous* means that multiple recruitment mechanisms were used. The row color signals the system's writing platform: **Custom System** (Gray), **Google Docs** (Apricot) [19], **Microsoft Word** (Green) [51], **Etherpad Lite** (Salmon) [63], and **Unknown** (Blue).

3.3 Interview procedures

Our goal in conducting interviews was to gain insight into the development of crowd-writing systems. As possible interviewees, we considered the authors of the papers selected for inclusion in the literature review, removing one paper from consideration due to a conflict-of-interest. We recruited authors for the interview study in two rounds. First, we contacted all 23 lead researchers[3] via email or an online messaging platform. Second, for each paper we tried to identify the author(s) who advised the system design (referred to as the "advising author"). We also employed a type of snowball sampling approach [8] during our correspondence with lead researchers, in which they directed us to an advising author. Through this process, we found that several lead researchers on early crowd-writing research are also advising authors on more recent studies. While we recognize the contributions of the 75 individual authors of papers in our literature review, it was not feasible to consider all of them for interviews due to volume and time constraints.

In total, we interviewed 18 researchers who collectively were authors on 21 of the 24 papers included in the literature review. Each interview was 20 to 90 minutes long and conducted via phone or video call. When possible, both authors of this paper interviewed an author together, otherwise interviews were conducted by a single researcher. The interview questions focused on the design decisions identified during our literature review analysis. We additionally asked questions about system implementation and goals for the research, as well as about their general perspective of crowd-writing research.

Interview responses were analyzed using a similar analytic memoing approach [61] as applied to the literature review. We both reviewed each interview by consulting transcripts, handwritten notes, and audio recordings. The initial review generated a collection of quotes, which we organized by consensus around common writing components in the cognitive process model [16]. Quotes presented below may be paraphrased, due to note taking inaccuracies or to remove identifying details. We randomly assigned each interviewee a unique identifier ranging from 1-30, which is represented in the text as ⟨PX⟩. Findings from the interview study were used to enrich and expand on design decisions identified through the literature review.

4 FINDINGS

Each subsection in the Findings presents one component of the cognitive process model [16] and, organized within it, are design decisions and insights based on the literature review[4] and interview study. While presenting the cognitive process model in written form naturally injects a linear ordering to the components, recall that the components represent elements of a potentially more complex, recursively-structured writing process. Tables 2 through 12 outline the design decisions identified for each component, alongside the citations for the papers in our literature review which exhibit that decision.

4.1 The Writing Assignment

The Writing Assignment consists of exactly what the solo writer is being asked to write, from a one-page essay about escalators to a 500-word ad for a used car in a newspaper. The writing assignment is the main puzzle a solo writer faces: they are asked to answer the assignment based on their own interpretation of what it asks them to do [16]. In the context of the cognitive model, clients set the assignment, but the solo writer is the one who shapes the written output.

Through our review, we found no system where crowd workers choose, or even influence, the writing assignment directly. A few systems, including *Mechanical Novel* [31], incorporate reflection

[3]This was the first author in all but two cases, in which we knew already that the lead was another author.
[4]In the Findings, we use *italics* to identify the few citations that are *not* part of the literature review.

processes in which crowd workers reflect on drafts of the writing. While the ability to reflect on a draft document enables a number of cognitive processes (e.g., Goal-setting, Monitor), having power over the writing assignment means making fundamental choices about which writing standards to consider and what prior knowledge may be relevant.

Our findings show that crowd-writing system designers choose writing assignments because they fit some external research question the designers want to answer. As described in Table 2, we observe three main categories of external research questions: the study of personal productivity, organizational social science, and machine learning. By considering the overall goal of a crowd-writing system through the concept of the writing assignment, we were able to identify these major cross-system themes.

4.1.1 Systems that enhance personal productivity. Writing is a core skill for almost all professions. Recognizing this, numerous crowd-writing systems are designed to support experts in a professional setting: "We might decide we want to support historians, journalists, or other types of domain experts and they do a lot of work with text" ⟨P11⟩. Several crowd-writing efforts integrate crowds into word processing systems in order to enhance the personal productivity of clients (e.g., *Soylent* [7], *WearWrite* [52]). Subsequent research has attempted to structure writing plans to promote a faster revision cycle [29].

The act of writing can also provide professional development more generally: "by forcing people to write text, you're forcing people to preserve learning," ⟨P16⟩ "it's about sharing with yourself at a later time" ⟨P26⟩. However, writing is hard and requires focused time, so people are not always in the right mood to write. Recent research has attempted to account for the mood of the worker: "you want a balance between making people do what they want and don't want to do" ⟨P8⟩.

A few interviewees discussed how future systems might enable people to express their ideas without "carrying them forward into writing themselves" ⟨P26⟩. Such a system might "maintain a North Star for the author's intent [...], but hand that end user intent off to a group to magnify it" ⟨P13⟩. This is a challenging technical problem, as "a very good piece of writing has a constant goal and understanding of the goal, but trying to pass that off from worker to worker in a short amount of time is very difficult" ⟨P3⟩.

4.1.2 Building systems to study organization-level social science questions. Research considered in this paper extends beyond computer science to areas such as psychology and organizational behavior:

- How might a crowd act with global understanding, when each contributor only has access to local views of the effort (e.g., [21, 31, 33, 42, 66])?
- How might we improve communication among partners who write together (e.g., advisor-to-advisee [29], editor-to-writer [57])?
- How might we enhance access to information (e.g., in data visualizations [72], online discussions [50, 74], humorous memes [43], student assignments [18])?

By easing many of the barriers involved with coordinating people in physical space, computing systems enable novel forms of human organization *[48, 58]*. Crowd-writing shares commonalities with organizational science: "I don't think it's that different ... maybe just scaled up in terms of the number of people and scaled down in terms of length of engagement" ⟨P12⟩. Crowd-writing systems are also used to prototype different ways of organizing people, "you're not interested in the technique being done by crowdworkers, but you're interested in the dynamics that happen with paid crowdworkers, and later want to perform the work with students, teachers, employees, who are harder to recruit" ⟨P11⟩. In this way, the purpose of a crowd-writing system might be to produce written output, but it might also offer some new view into human behavior.

Table 2. Writing Assignment (§4.1) Design Decisions

	Design Decision	References
1	Enhancing personal productivity	[7] [29] [52]
2	Social science: global outcomes through local actions	[21] [31] [33] [42] [66]
3	Social science: improve communication among writing groups	[29] [57]
4	Social science: enhance access to information	[18] [43] [50] [72]
5	Training machine learning algorithms	[21]

4.1.3 Training machine learning systems. Online labor markets, like Amazon's Mechanical Turk (AMT) [4], have helped to advance machine learning research: "AI systems are working thanks to crowdsourcing, these systems are successful because we have a lot of labeled data" ⟨P5⟩. However, it is less clear what advancing artificial intelligence (AI) systems means for the future of crowd work: "Increasingly we're seeing crowd workers as part of an AI pipeline, where if the classifier does not work, then the failure is passed to crowd workers to evaluate—that's interesting, but it's important to pay attention to what are the capabilities of NLP, AI, etc., because that's changing fast, and AI is now often better than crowds" ⟨P11⟩.

While some interviewees shared that they view AI systems as outpacing crowds, coordinating humans to perform a complex task can be fairly different than programming a computing system to perform the same work. As explained by ⟨P13⟩, "because we are summarizing with crowds, we recognize problems that don't emerge in automated summarization. A lot of times using human intelligence to solve problems, lets you think farther than AI [...] confront[ing] other challenges, because we are working with people, groups, and crowds."

Several interviewees ruminated about whether there is a future for crowd-writing systems research. A segment of crowdsourcing research falls into a pattern that one interviewee characterized as, "*look we can use crowdsourcing to do X*, but we've seen 50 papers on that topic, no one cares about that now" ⟨P20⟩. Similarly, "the research community may not have much more appetite for this type of research, either we've solved all of the problems, or it was just a fad [...] maybe it's time to use the techniques that we have learned building those systems to solve real problems" ⟨P5⟩.

4.2 Topic

Solo writers need to understand what they are writing about [16]. To do so, they typically prepare to write by researching source materials related to a writing topic. Similarly, crowd-writing systems incorporate processes to discover and analyze source materials (e.g., historical artifacts [68], social media [30], data visualizations [72]). Examples in this section center on task design as well as how crowd-writing systems call these tasks redundantly to discover and analyze information from source materials (Table 3).

When considering a solo writer, the effort of discovering and analyzing source materials is typically subdivided into tasks that can all be completed in sequence by the writer. Crowd-writing systems typically accomplish this task instead by subdividing the effort into tasks that are coordinated among a crowd of people working in parallel. For example, the first micro-task in the *Knowledge Accelerator* [21] workflow for discovering source materials involves hiring five workers to issue queries about a writing topic to a search engine and then return the top five results. During our interviews, researchers with experience applying this variety of task spoke positively about its value, because "it builds on the success of the Wikipedia citation model" ⟨P24⟩ and "references are also easy to check" ⟨P14⟩. Hiring multiple workers to perform the same task in parallel can help validate references, assuming that a source returned by two or more workers is likely relevant [21].

Table 3. Topic (§4.2) Design Decisions

	Design Decision	References
6	Finding source material via multiple parallel tasks	[21] [68]
7	Handling difficult-to-anticipate outcomes using redundant source material	[21] [30]
8	Teaching skills while sorting through specialized source materials	[68]

Adding redundancy, by hiring additional workers to perform the same task, can be useful when the outcomes are hard to anticipate or to evaluate objectively.

After identifying a collection of candidate sources to review, crowd-writing systems coordinate workers to sift through the materials for facts to incorporate into the writing. The Knowledge Accelerator workflow involves hiring two workers to review each source for five snippets of text; however, in trials of the system, workers often focused all of their attention on the beginning of the source material. Hahn et al. [21] discuss how the instructions and user interface for a micro-task can be used to motivate workers to spread their effort across a source, by highlighting sections that are already identified by other workers. With each additional snippet returned, the Knowledge Accelerator hires another two workers to review the source—in this case, redundancy is useful because the amount of information that a source contains is hard to anticipate.

Once separated from the source material, however, workers can struggle with the context for a snippet. As an interviewee shared,

> "So a sentence that tortured us was: *The Connecticut Huskies took the championship this December*. You want the most specific content. If you do the reading it's for the U. Conn. Women's team for a specific year. There were enough of these [torturous sentences] that the problem was ambiguous" ⟨P24⟩.

In this case, crowd workers had to rely on their own familiarity with New England athletics or a search engine to recognize the context.

Redundancy can also play a role in analyzing relevant source materials once discovered. Typically, solo writers generate questions about a topic as they analyze source materials [16]. In crowd-writing, *Storia* [30] provides a slight variation on the process of extracting facts from source materials, by hiring workers to initially generate questions that a typical viewer might ask when viewing the source, and then hiring another group of workers to answer the questions, by reviewing source materials or querying a search engine. In this case, redundancy is useful because the number of question-answer pairs per source is hard to anticipate.

Special skills are needed to interpret some source materials. Rather than hire specialists as a solo writer might, some crowd-writing systems include opportunities for workers to develop these skills in-situ. For example, the micro-task scaffolding designed into the *CrowdSCIM* platform [68] trains workers in "historical thinking" based on the SCIM-C method *[22]*, which they practice by performing micro-tasks towards analyzing historical artifacts. While a worker may not develop a deep understanding about a writing topic through micro-task work, CrowdSCIM demonstrates that micro-tasks can offer opportunities for workers to develop expertise in critical aspects of the writing process, such as how to interpret source materials.

Overall, Topic-related designs suggest productive avenues for future system designers. Existing systems succeed by exposing workers to the Topic while searching for source material and confirming their findings by asking other workers to do equivalent, redundant tasks. These principles are likely to be specifically useful for future designers who wish to consider complex topics or determine the topic during system execution.

4.3 Audience

Solo writers write for an audience. They develop an understanding of their audience as they research and write about a topic, but it is less clear how information about the audience is integrated into the moment-to-moment act of composing [16]. Examples in this section demonstrate how the proximity of a crowd worker to the audience for a writing topic can enable them to play an active role in the process of writing for an audience (Table 4). Indeed, in crowd-writing systems, Audience can have a strong influence on composing.

Some systems ask workers to help clients iterate on their own writing by providing feedback. Sometimes, workers serve as an intermediary audience for the writing [46, 73]. Micro-tasks that involve providing feedback ask workers to shift from performing assistant-like tasks, like discovering and analyzing sources [21, 30, 33], to more editing tasks, where they can provide potentially valuable opinions about how an audience may perceive the writing [46, 73].

Learning about an audience also means learning their language, such as jargon and idioms, so that a solo writer can communicate about a topic in familiar terms. Some workers perform this process as well, by identifying possible points of confusion for an audience in the Text Produced So Far [30, 43]. Lin et al. [43] offer an example of this in the context of crowd-writing to generate explanations of humorous English-language memes for non-native English speakers (see also *Hong et al. [23]*).

A writing task may relate to a worker's personal experiences, interests, or beliefs. In some cases, a worker may be unable to separate their interest in the topic from their contributions to the writing. Several papers in this review present evidence that, when a worker has an opinion about the writing topic, particularly controversial topics, their performance is lower than that of workers who feel differently [50] or who may be less invested in the topic [74]. During our interviews, an interviewee shared a relevant anecdote about a crowd-writing system feature that they tried, but abandoned:

> "I would write out notes on what I was trying to say and see if I could bring in help to flesh out the sentences. But, it became clear that I was not effectively communicating intent. As one example, I started with a sentence against drilling in Alaska, and asked people to flesh this out, but what came back was very pro-drilling. So, I focused on other parts of the software where intent could be more clearly communicated" ⟨P29⟩.

Rather than design ways to minimize the influence of a worker's opinion, Mahyar et al. [47] discuss how crowdsourcing may empower a community to generate ideas directly from their opinions. However, this is a delicate balance. A worker might not only have an opinion about a topic, but sharing their opinion as part of crowd-writing may have negative consequences. Mahyar et al. [47] discuss the tension between protecting the privacy of their workers, members of a neighborhood with a shared concern about an urban planning decision, and requests from city officials (audience members), who wanted workers' names and demographic information.

The tension between crowdsourcing and a worker's proximity to the topic, can be even more pronounced with crowdsourcing in real-world contexts. Agapie et al. [1] discuss the social challenges involved with assigning workers to report from their own community to coordinate local event-reporting, including that workers may feel separated from their community when performing tasks. These community-driven examples, underscore a lesson for crowdsourcing in real-world contexts: pay attention to the workers, volunteers, as well as the needs of the community where the crowd work is situated. This observation is also discussed in existing research about group awareness in collaborative writing *[45]*.

Table 4. Audience (§4.3) Design Decisions

	Design Decision	References
9	Workers can serve as an intermediate audience	[46] [73]
10	Workers identify potential sources of confusion	[30] [43]
11	Workers may provide their own perspectives on the topic	[1] [47] [50] [74]

4.4 Long-Term Memory

The Long-Term Memory is a solo writer's "storehouse of knowledge" [16, pg. 371], which includes a solo writer's own understanding about how to write for a specific genre (e.g., gardening) as well as their personal understanding and beliefs about a writing topic, such as how to grow better squash. The cognitive process model [16] describes two challenges related to Long-Term Memory: (1) Retrieving information from memory about how to write for a genre or a topic, and (2) Determining how to apply the retrieved information to a specific writing task.

While some crowdsourcing systems recruit experts *[56]*, many crowd-writing systems coordinate micro-tasks, which can be performed by anyone with (at least) a general ability to write *[32]*. Designers appear to compensate for this difference in assumed skill level by incorporating a process of exposing workers to or reminding them of expert-level processes through task instructions and activities. In this section, we discuss how externalizing expert-level knowledge, in such a way, can afford crowd-writing processes that rely less on the memory of each worker and more on in-situ responses from workers (Table 5).

Many existing systems draw task design inspiration from known strategies for writing. Writing strategies, as defined by *Greer et al. [20, pg. 3]*, are "semi-rigid workflows that [...] decomposes the complex task of writing and transforms it to smaller, more manageable subtasks." As an example, two papers [52, 57] apply a variation of the following workflow to generate a paragraph: (1) Create a list of relevant sub-topics, (2) Order the list to introduce each sub-topic in sequence, (3) Compose a paragraph based on the ordered list of sub-topics, and (4) Improve the paragraph until it is complete. This workflow reflects one of several high-level generic strategies which can be applied to a range of genres; Section 4.6 addresses executing this type of workflow as well as fine-grained strategies.

Strategies from specific genres also inform system workflows. As mentioned above, CrowdSCIM [68] adapts the SCIM-C educational workflow into five micro-tasks each performed by a single worker. Storia [30] applies writing standards from narrative fiction to summarize events from a collection of social media posts, where workers use four narrative categories (i.e., establisher, initial, peak, release) *[11]*. Such tasks can also be applied repeatedly to improve a piece of content, as demonstrated by *CrowdIA*, which recruits workers to perform cycles in a series of sensemaking loops *[54]* that iteratively refine an analysis of a mystery [42]. In each case, the long-term memory of how to write is externalized as writing task activities that workers respond to in-situ, which is quite different from the way that solo writers rely on their own long-term memory of known writing strategies.

Flower and Hayes [16] emphasize that all solo writers, including subject-matter experts, struggle to access their own long-term memories about specific topics. To address this, some crowd-writing systems expose workers to topic-related content and then record the workers' immediate responses. The subsequent record of worker responses functions as the system's memory, which can be accessed and used later in the system pipeline. For example, Glassman et al. [18] found that students in a computer architecture course will collectively learn a lot about the process of error-testing bugs, but a student may not remember each of their solutions to each bug as time passes. Glassman et al. [18] build a system to resolve this issue, which asks students to generate hints about

Table 5. Memory (§4.4) Design Decisions

	Design Decision	References
12	Externalizing tasks with general workflows	[52] [57]
13	Externalizing tasks with genre-specific workflows	[30] [42] [68]
14	Generating a collective "crowd" memory with workers' impressions	[15] [18] [38] [75]

how they solved bugs while they are fresh in their mind. The hints are then stored in a collective database, providing a system-wide memory for other students in the course.

A system's memory can also be structured to prioritize information related to a specific writing topic. A useful example is *Chorus* [38], a crowd-powered chat system designed to facilitate real-time responses to client questions (e.g., where to eat when visiting a new city). Rather than using experts, Chorus elicits numerous responses from multiple workers, then ranks and sends the highest rated response back to the client. As the client responds and asks subsequent questions, workers add notes about the client into the system's shared memory, so that even newcomers to the task can generate responses that address known client needs. Chorus's shared memory helps the crowd to maintain conversational continuity with the client [38] in a similar way to how the hint database described above helps students.

In contrast to a solo writer, who queries their own memory, crowd-writing systems can recommend specific content from the system's memory for a worker to consider. Personalized recommendation can be useful in crowd-writing tasks that ask workers to actively reflect on their own perspectives on a topic [15, 50]. Drapeau et al. [15] in *MicroTalk* ask workers performing a linguistic relation extraction task to reconsider their task response when a prior worker responded with a different answer (described as *asynchronous deliberation*). All of the above examples demonstrate how a system can construct a long-term memory of a writing topic, by eliciting responses from workers when they are exposed to relevant content.

Supporting shared memory systems in crowd-writing can be challenging and cause unexpected behavior that likely would not occur in the solo writer's memory. For instance, "The first time we built the memory tool it was an open chat box between workers, it would motivate long term type tasks[. T]hat failed fantastically, because having that open of a channel, people ignored the main task, treating the interface like a water cooler [at work]" ⟨P13⟩. Explicit workflows for crowds are likely better at structuring the process of retrieving and applying information from Memory than the average solo writer.

4.5 Planning

Solo writers execute Planning to build their "internal representation of the knowledge that will be used in writing" [16, pg. 372]. Practically, they take their knowledge of the writing topic, audience, and assignment and transform it into their actual writing plan during Planning.

Planning consists of three sub-components: (1) Generating ideas, (2) Organizing, and (3) Goal-setting. The solo writer uses these sub-components to refine and structure how to produce the written output. While executing the work related to these sub-components, solo writers see the writing through multiple perspectives, from local understanding of specific sentences and paragraphs to a global view of the written output (e.g., narrative argument, plot). Given they are writing alone, solo writers can navigate between these views whenever they would like. In contrast, crowd-writing systems must determine how to communicate the global view to workers while they perform generating, organizing, or goal-setting tasks at a local level.

4.5.1 Generating Ideas. Generating is an "elementary mental process" [16, pg. 367] that a solo writer uses to convert global information into actionable ideas. By and large, generating an idea involves consulting the writer's memory and pulling out concepts that may be relevant to the task at hand. Sometimes these ideas are well-formed, yet other times they may be loosely structured [16]. By contrast, the process of generating ideas in a crowd-writing system is highly structured, with multiple sub-processes.

Generating an idea with the crowd frequently consists of partitioning existing content about a writing assignment and topic into chunks suitable for micro-task work (Table 6). Deciding how to partition the content is not an easy task. If the order of the content is meaningful, as in a novel, then there are two primary considerations: "how large the chunks should be and the other thing is where you should split" ⟨P4⟩. If the content order is less meaningful, it may be useful to curate each chunk, "we [chunked] the content by describing the fixed sizes, 1, 3, or X, and also deciding which content to put into each [chunk], by using [an automated method] to put relevant [content] together" ⟨P6⟩. Some crowd-writing systems ask workers to partition the content. This can be a source of brittleness though, as "[t]his creates the possibility that a single bad partition (i.e., outline) could have a large negative effect on the whole task" [33, pg. 46].

Systems use a variety of methods to partition content, involving workers [33], machine learning [30, 43, 66], or a hybrid process [21]. Some partitioning tasks require contextual knowledge, such as determining criteria for buying a car [33] or what comments are directly relevant to others in a discussion [74]. Systems which involve workers in the process of partitioning content can be surprisingly robust [33, 42]. CrowdIA's evaluation suggests that crowds can remove noise when partitioning a large quantity of information without losing important data [42]. *CrowdForge*, a system which asks workers to research and write articles on decision-making tasks, found that even if some workers classify incorrectly at the beginning of the task, the crowd still has more overall benefit than an individual working alone [33].

Some partitioning tasks use voting or paired comparison to rank the partitions [18, 66]. As each partition represents a local view of the content, the resulting rank order may not adequately reflect the global context. A solution discussed by interviewees is to create an initial ranking and then ask workers to review each partition with context: "sometimes students would look at the first [response] and it made no sense, then the second and it made more sense, then the third and they got it" ⟨P16⟩. A more formal workflow discussed by ⟨P4⟩ is to use paired comparison tasks to construct an initial hierarchy of partitions and then, "use the hierarchy to go down, backwards through the structure to improve the ratings."

Numerous crowd-writing systems in this review use machine learning methods to parse content. Examples include clustering social media posts [30], segmenting meme text into how it is used comedically [43], and splitting book or movie scenes using scene detection algorithms [66]. Systems use automated methods as a first pass towards sorting ideas or information and the output of this pass then meets the algorithm's specifications. As a crowd-writing system is already a computational system, it is likely easier to augment a writing process with machine learning than to augment the practice of a solo writer in such a way.

The Knowledge Accelerator introduces the *open-ended set sampling* design pattern to partition large datasets, while providing workers with a sense of the overall context. Open-ended set sampling asks workers to repeatedly sample the source material to form distinct groups or categories. This process continues until workers believe they have identified a set of distinct groups. Information about the distinct groups is then used to run an automatic method (SVM classifier), and subsequently tasks additional workers to filter the classifier's output. As part of the final step, the system tasks workers to familiarize themselves with the automatically generated groups by writing short

Table 6. Planning - Generating (§4.5.1) Design Decisions

	Design Decision	**References**
15	General partitioning of content with workers	[33] [74] [42]
16	Partitioning data using voting or comparison methods	[18] [66]
17	Machine learning-based content partitioning	[30] [43] [66]
18	Open-set sampling: hybrid (worker & ML) partitioning	[21]

descriptions of the evidence contained in each group. This local insight from workers at the seed and filtering stage enhances the system's global organization of content [21].

Crowd-writing system design decisions related to generating ideas are highly structured, and lack the "fragmentary, unconnected, or even contradictory" [16, pg. 372] ideas which may arise for a solo writer.

4.5.2 Organizing. Once solo writers refine their ideas extractively or abstractively, they make meaning and provide order to the ideas through Organizing. There are two main types of organizing in the cognitive process model: (1) Organizing writing textually by setting minor goals or focusing on the shape of the text, and (2) Organizing writing conceptually by categorizing ideas, determining sub and super ideas, and developing new concepts [16]. Solo writers navigate between global and local views of the written output while organizing. This suggests similar global-local challenges for crowd-writing systems; our analysis found that crowd-writing systems have been designed to address these challenges by tasking workers to respond to structured prompts and recursively summarize writing task responses (Table 7).

Some crowd-writing systems organize ideas around specific questions [1, 21, 30]. Asking the crowd to generate answers to a specific question is similar to how a news desk editor dispatches reporters to an event, as described by Agapie et al. [1, pg. 6], "Could you ask Paul a question for me? Ask him what is his favorite thing to draw on his tiles," and "Did he [the speaker] specify which [...] executive that was?" All of the generated answers can then be reduced into a synthesis that responds to a specific question; a process which ⟨P13⟩ described as *aspect summarization*:

> "For example, a common summarization task is how do you translate the play-by-play reports from baseball into just the scoring drives, but that assumes that you care about score [...] you might care about injury statistics or updating your fantasy baseball team[. In this way] summarization is not a monolith, we often summarize specific aspects of text to get specific outcomes" ⟨P13⟩.

Summarization tasks involve workers in multiple levels of the content. For example, *Context Trees* [66] coordinates workers to summarize a story into a tree-like structure. The Context Trees approach involves two general tasks: (1) Tasks that combine lower levels of the information about a story into increasingly higher level summaries and (2) comparison tasks, in which workers rate the summaries at each level of the tree. The Context Trees algorithm then uses the worker ratings to move each summary up and down the tree, in order to promote better content to the top. Rather than using worker ratings of the content to navigate a tree, *Wikum*, a more recent platform, provides workers with an interactive representation of the tree they can navigate themselves [74]. In this way, both crowd-writing systems ask workers to recursively consider local aspects of the written output in its broader context (global level).

Summarization is a difficult crowd-writing task because humans are in the loop. ⟨P3⟩ said that some workers struggle with analyzing summaries written by others. They note that workers would prefer working with writing that has gone through the worker's own "brain filter" first. Additionally,

Table 7. Planning - Organizing (§4.5.2) Design Decisions

	Design Decision	References
19	Answering explicit questions as prompts to generate ideas	[1] [21] [30]
20	(Recursively) summarizing information to further synthesize ideas	[66] [74]

summarization tasks in crowd-writing raise some socio-techical questions about what it means to follow your contributions as they are integrated into a summary:

> "[I]s it OK for people to break up discussions and move pieces from one thread to another? [...] While it is possible to move threads in [the crowd-writing system], people did not use it in the trials" ⟨P27⟩.

While crowd work often involves independent micro-tasks, summarizing a discussion with crowd workers may require more group awareness.

4.5.3 Goal-setting. Goal-setting tasks move the writing process forward as a solo writer's global view of the information evolves. Setting specific goals can happen for different aspects of the text and at different levels of specificity. Solo writers typically focus on two aspects of the text, the content and the process, and set goals for each [16]. Content goals guide solo writers to add material to the text, for instance expanding on a topic or clarifying something confusing. Process goals address the mechanics of generating text, such as adding a paragraph or spell checking [16]. At the same time, the level of specificity of the goals can vary as well: sometimes goals are global to the full text at hand or local to a specific sentence or even word.

While Goal-setting plays a role in all crowd-writing systems, our analysis found that this role is typically implicit in task design and worker instructions. Some crowd-writing systems include design components that help workers set primarily global content goals [29, 31]. Some systems integrate process goals and writing mechanics, mostly when a micro-task asks them to generate text (Translating) [49]. Other systems do not ask workers to be part of Goal-setting, but rather assume that a client will set/adjust the goals [57, 72]. Still other systems seed goals using machine learning-led Organizing [21]. Regardless of where the goals originate, determining how to communicate goals effectively to the workers is a key challenge for crowd-writing systems design [31, 72].

Kim et al. [31] and Kaur et al. [29] present crowd-writing studies that directly design for Goal-setting. The Mechanical Novel [31] generates multiple scenes for a short fictitious story, by coordinating two workflows: one for reflecting and another for revising. In the reflect workflow, the system identifies a set of global goals, by first asking a group of workers to critique scenes in the story, and then recruiting another group of workers to vote for critiques that should be adopted. In the revise workflow, the system locks and unlocks scenes in the story for editing, based on the number of votes received for each critique, and then recruits a group of workers to resolve the most important critiques by editing the unlocked scenes [31].

Rather than asking workers to define each critique, Kaur et al. [29] present a scaffolded approach based on a pre-defined list of primitive functions (called a *vocabulary*). With such a pre-defined list, workers are able to assemble step-by-step plans to accomplish specific writing goals. Both approaches highly constrain how workers can set global goals, with a locking structure [31] and fixed vocabulary [29], respectively. By contrast, a solo writer has considerably more agency over goal-setting.

4.6 Translating

Translating is the process of taking plans and turning them into sentences, paragraphs, and ultimately a written output. Converting the output of Planning, plus anything from the writing

Table 8. Planning - Goal-setting (§4.5.3) Design Decisions

	Design Decision	References
21	Asking workers to set global content goals	[29] [31]
22	Determining how to communicate goals to workers	[31] [72]

Table 9. Translating (§4.6) Design Decisions

	Design Decision	References
23	Asking workers to produce generic writing output	[1] [7] [30] [52]
24	Requiring workers to meet special/specific output formats	[1] [7] [21] [30] [47] [72]
25	Adapting writing standards from existing genres to the crowd	[30] [50] [68]

assignment, topic, or audience, into text is a significant effort for the solo writer [16]. Solo writers perform a sort-of balancing act when composing, which can increase their cognitive load. They must "juggle" [16, pg. 373] writing mechanics, such as grammar and spelling, while determining how to convey their ideas through text.

Our analysis found that crowd-writing systems must also manage different tasks as part of Translating. Systems handle this challenge by hiring workers to perform three categories of specific, structured tasks: (1) Generic tasks, (2) Tasks that respond to specific output formats, and (3) Tasks modeled after a genre's writing standards (Table 9). These tasks address different parts of the writing and workers typically complete each type of task independently.

Generic writing tasks reflect the basic sequence of work needed to generate a specific written output (e.g., paragraphs [7, 52], summary of an event [1, 30]). For example, WearWrite [52] defines several generic tasks that coordinate workers to expand bullet point statements into sentences, sentences into paragraphs, and to further refine and organize the paragraphs into a coherent output. By completing these generic tasks, ideally workers are able to improve a document without additional instruction. The recipe for generic steps necessary to construct a basic written output depends on the text's format. For example, writing that involves news listicles [1], non-text items [30, 47, 72], or captions and reference lists [7, 21], require some similar and different generic tasks.

Many crowd-writing systems adapt writing standards from a specific genre. These genre-specific scaffolds are similar to the writing strategies from Memory, but act at a more granular stage of writing, informing the construction of sentences and paragraphs. For instance, the system presented by McInnis et al. [50] tasked workers to write discussion prompts, providing them with a scaffold called *Steps to Writing a Good Discussion Prompt*. Both Storia [30] and CrowdSCIM [68] also use writing guidelines from their respective genres of narrative structure and historical analysis. Systems that target specific genres not only restrict workers to a specific Topic, but also codify the writing standards for the genre.

4.7 Text Produced So Far

Through Translating and the other writing processes, a solo writer produces text. This text, described as the "Text Produced So Far" or the growing text, influences how the solo writer navigates other processes [16]. As many solo writers make decisions based on what they have already written [16], the growing text's very existence constrains what next steps a solo writer can take. For instance, if the text is becoming unwieldy, they may begin Reviewing or, if they feel the text is incoherent, they may initiate a round of Planning. Additionally, solo writers view the growing text as an artifact that they can query at any time to evaluate their writing progress [16].

Table 10. Text Produced So Far (§4.7) Design Decisions

	Design Decision	References
26	Passive Presentation: Giving specific text without requiring engagement	[30] [46]
27	Active Presentation: Requiring workers to interact with the text	[21] [66]
28	Role-based Presentation: Giving specific text according to worker role	[1]

While solo writers enjoy full access to the growing text, by contrast workers in crowd-writing systems frequently have minimal access. A crowd-writing system may only present workers with a limited view of the growing text, which can be just the text that the worker has produced. This fully distributed approach is a challenge for system design: "It was clear early on that it would be hard to write something where no individual knows the whole structure" ⟨P29⟩. In order to provide workers with global context for their local writing tasks, crowd-writing systems apply a variety of approaches to manage when and how much of the existing text to present. Below we highlight three approaches identified by our analysis: i.e., passive presentation [30, 46], active presentation [21, 66], and role-based presentation [1] (Table 10).

Passive presentation occurs when the system presents a worker with only the sections of the growing text that are relevant to their individual task. For instance, as an individual worker starts a task in Storia, they are presented with paragraphs from the growing text at the top of the interface and asked to skim them [30]. Luther et al. [46] adapt progressive disclosure *[60]* to explore asking workers with different levels of expertise to critique student posters according to seven design principles. Progressive disclosure allows information about the design principles and the growing text to be hidden, in order to not overload the worker, while also giving them an easy way to refer back to the material. These passive presentation designs encourage the worker to consult the growing text, but do not require them to engage with it.

Interviewees described challenges underlying how to present workers and clients with the text to read. Both ⟨P11⟩ and ⟨P16⟩ noted that reading text can be difficult compared to other mediums. ⟨P11⟩ said that an "image is self-contained [...] it seems more manageable for workers than reading multiple pages of text." Beyond reading, ⟨P20⟩ noted that infrastructure can impact one's ability to interact with the growing text. For instance, small screens or ill-defined interfaces can restrict the text and in turn the client's or worker's understanding of it. Another consideration is whether workers are able to view contributions to the text made by other workers. In some contexts, it is useful to keep these contributions separate, "we wanted several people to be working in the same document [but] we didn't want the student to see [it ...] until we showed it to them, so that collaborative aspect was challenging" ⟨P8⟩.

Active presentation systems engage workers directly with the growing text. Both Context Trees [66] and The Knowledge Accelerator [21] present snippets of the growing text to workers at the beginning of the task, similar to Storia's approach described above. However, they both add an additional evaluation step. In Context Trees, a worker first reads summaries from lower levels of the tree and votes on the most informative, prior to summarizing them. The Knowledge Accelerator uses the *evaluate-then-act* pattern which asks workers to read existing information provided by other workers and then provide evidence for or against it. These methods guide workers through reading complex text by adding an active evaluation component.

Role-based presentation divides workers into writing roles and provides them only with text associated with that role. As a simple example of assigning roles, a designer could assign workers one role for each component of the cognitive process their system addresses: e.g., reviewer, translator, monitor, etc. Not all systems use roles explicitly, but some designs [14, 52] have some role-based structure or elude to it. The system described in Agapie et al. [1] replicates an existing role

breakdown from journalism (e.g., reporter, content curator, writer, workforce manager). The reporter, who is attending events in real-time, only generates content and has no access to the growing text. In contrast, the writer individually produces the growing text based on this content and thus has full access to it. This differs from collaborative writing, where, even though solo writers may have roles, they each individually can maintain full access to and agency over the text.

While crowd-writing systems may task a worker to play a particular role in a workflow, it is important to remember that the people recruited for this work may have relevant background (e.g., expertise, skills), that may or may not be well suited for their assigned role:

> "[T]here was a professional copy writer who was part of the process and was getting very frustrated by the less skilled writers in the crowd. You're being asked to collaborate with people you don't know, you've never met before" ⟨P20⟩.

Unlike collaborative writing, a crowd-writing system might recruit people who have never worked together before to execute roles that may be new to them. Exploring whether workers resonate more with their own backgrounds or their assigned role, and further how to encourage them one way or the other, may be useful future work.

4.8 Reviewing

The recursive nature of the cognitive process model reflects the fact that the solo writer can always change the growing text. The Reviewing process is where the solo writer assesses if changes need to be made and then executes those changes [16]. For example, this process can initiate a cycle of Planning by referring to existing writing guidelines in Memory. Throughout the writing process, a solo writer will use their own writing goals (and any guidelines set by their client) to evaluate the growing text, as part of their own internal monologue about the writing [16].

Our analysis found that typical crowd-writing Reviewing tasks are either generic or targeted. Similar to generic Translating tasks, generic Reviewing tasks can provide value to the worker and improve the writing (e.g., spellcheck, grammar check). Targeted tasks are intended to address a specific critique about the writing, such as "write two sentences about why a non-computer scientist would care about this work," [52, pg. 3839] or "the reports [are] too neutral [...] and need to be tweaked to meet the needs of the audience" ⟨P5⟩. In this section, we discuss how crowd-writing systems manage Reviewing tasks that are generic, in that they reflect standard steps that a solo writer would take to refine their writing, as well as how crowd-writing systems manage tasks that are specific to the growing text. Reviewing can also involve coordination between tasks, which some crowd-writing systems implement with a shared to-do list (Table 11).

A common generic reviewing task is to identify and remove duplicate material from the growing text. Work by Willett et al. [72] asks workers to review multiple crowd-generated explanations of a data visualization and then mark at most one other explanation as redundant. The system uses those assessments to automatically find a set of unique representations using a search over a graph of noted redundancies. Storia implements a similar worker-based voting system to remove redundant paragraphs from their event summaries. Notably, the redundancy present in Storia originated with its source material (Seen.co event clusters), pointing to the more general crowdsourcing-scale challenges crowd-writing systems face [30].

Systems also ask workers to perform generic tasks related to the coherence of the growing text, such as by reorganizing the order of paragraphs or revising a paragraph to reflect the style of writing in other paragraphs. Such inconsistencies in the writing may exist because workers have a limited view of the growing text (e.g., passive presentation, role-based presentation). One method of resolving inconsistencies is to facilitate edits both within and between subtopics. The *vote-then-edit* pattern interleaves within and between workflows, allowing for repetitive refinement

Table 11. Reviewing (§4.8) Design Decisions

	Design Decision	References
29	Structuring remove of redundant text or content from the growing text	[30] [72]
30	Vote-then-edit: improving text coherence through reviewing	[21]
31	Facilitating communication of reviewing tasks from client to worker	[57] [72]
32	Communicating reviewing tasks via shared to-do lists	[18] [47] [52]

of an article while managing the different context burdens of editing in the two styles [21]. Overall, an advantage of generic reviewing tasks is that once workers are trained to perform a certain type of task, they can generalize it and then apply it to future writing.

Targeted reviewing tasks are more challenging than generic tasks to design for, because workers need to understand the writing context before they can address a specific critique [57]. Salehi et al. [57] present methods of structured communication for clients to communicate the writing context for targeted reviewing tasks to workers. Successful strategies include highlighting strengths and weaknesses in the text, comments and edits, and distilling the main problem with the current text; however, some communication strategies are better than other strategies at different stages of the writing process [57]. As another example, the data analysis workflow presented in Willett et al. [72] facilitates a conversation between the client and a crowd of workers, where the client is able to state what aspects of a data visualization need elaboration. In both cases, the client is directly involved with each Reviewing task in the crowd-writing system, by contrast to how a solo writer may receive feedback intermittently [16].

However, the client-to-crowd communication around Reviewing tasks can become overwhelming for client(s), particularly when managing a large crowd workforce. An interviewee discussed the challenge of determining how much information clients need about crowd edits to the growing text, "if you get a notification for each [small] edit, you'll be swiping all the time, and [...] eventually will auto-approve every edit regardless" ⟨P20⟩. To address this challenge, the interviewee's design team opted to create a threshold based on the amount of words changed in a suggested edit to classify edits that likely need a client's approval from those that may not.

During Reviewing a crowd-writing system may need to coordinate multiple tasks, in parallel, in series, and at multiple levels of the growing text. Several crowd-writing systems do this via a shared to-do list. Systems use to-do lists to allow workers to access and request Reviewing tasks, similar to the reflect stage of Mechanical Novel [31] described in Goal-setting. In WearWrite [52], generic revision tasks are always available to workers in a dynamic task queue. The review workflow in CommunityCrit [47] is similar, in that commenting on another worker's idea is always an available task. A unique feature of the educational system presented in Glassman et al. [18] is the ability to request reviewing from other workers. The system interface provides a button where students can communicate their need for new or better hints, which adds a task to the system's overall to-do list. Ultimately, these systems give workers the agency to request revisions themselves.

4.9 Monitor

Solo writers regularly shift their attention from one writing process to another: they might plan, then translate, review, and then plan again [16]. Many considerations can inform a solo writer's writing style, for instance their skills, time, or how they want to use their effort. Monitor reflects the constraints imposed by the writer's style and their writing progress by determining whether to remain or switch processes. The writer's constraints may change over the course of writing; the Monitor process, or the solo writer's writing blueprint, helps them react appropriately. Monitor processes help guide small writing decisions, such as whether to switch paragraphs or write another

sentence, rather than large writing decisions, like those outlined in the general writing strategies stored in Memory.

Crowd-writing system designers pay attention to how to allocate and represent constraints to workers and within a crowd-writing system when designing around Monitor (Table 12). The resources a crowd-writing system has for writing, such as how many workers are available, how many tasks can be assigned per-worker, and how are task outcomes coordinated among workers, are important constraints that are managed through Monitor processes. Some resource constraints are static (e.g., task time available, task reward amount). Other resource constraints are dynamic, such as the processes by which the system passes information from worker-to-worker (called *simultaneity constraints [48]*). While a solo writer may shift these constraints at any time, our analysis found that many resource constraints are fixed, as static system variables.

Hard-coding resources like time and money directly into workflows is a simple approach to resource management. For example, both CrowdIA [42] and Soylent [7] institute timeouts as part of multi-step workflows that generate complex written output. These timeouts aim to balance moving the workflow forward with getting more responses from workers; when worker response rates slow, time to completion could sharply increase without a timeout. MicroTalk is unique in encoding a financial budget directly into its algorithmic workflow for argumentation. Rather than using another end condition (e.g., elapsed time, number of responses), MicroTalk's workflow runs until a specific budget has been spent and no further [15].

Several crowd-writing systems use the character length of a written response as a static quality threshold, but also use this threshold to identify workers with higher or lower performance. Crowd-writing systems that control the length of the composed text may restrict it directly. Numerous systems specify the maximum number of characters allowed in a text box (e.g., [50], [74]). Length appears as a quality metric in a variety of systems because "it seems to indicate effort" ⟨P24⟩. In juxtaposition, clients tend not to put specific restrictions on solo writers in the same way. These examples show how monitoring resources can be static in crowd-writing systems.

Rather than depend on static constraints, some systems allocate resources dynamically. Dynamic crowd-writing systems respond to input from workers to make decisions, such as which sections from the growing text to continue revising, when to stop revising, and when to hire additional workers to perform additional tasks. One reason to build a dynamic system is because the content being written affects how much work needs to be done, as in answering How-To questions [21] or when facilitating a Question and Answer session [38].

Dynamic systems elicit feedback from workers about content through a variety of tasks, such as voting [31, 38], pair-wise comparison [21, 30], asynchronous deliberation [15], or by offering workers options about the type of tasks they would like to work on in the moment [47, 52]. However, these mechanisms for eliciting feedback are not flawless. For example, voting systems can be useful, but they depend on how questions are framed, the thresholds used for decision-making, and the sample of people included in the vote:

> "When the question is easy, majority vote always works. You can use expectation maximization to weight towards the people who are more correct. If no one has the right answer, you're probably [in trouble], but what if one person has the right answer and they are just outvoted?" ⟨P24⟩.

The *task of least resistance* design pattern attempts to consider both system resource constraints and worker effort by giving workers a choice. The system asks workers to choose what to work on from a number of options (e.g., versions of a paragraph to improve) and then, in a second stage they perform the task, benefiting from their choice, "in terms of having to do less work, easier

Table 12. Monitor (§4.9) Design Decisions

	Design Decision	References
33	Fixing resources such as time and money	[7] [15] [42]
34	Restricting length of the growing text	[50] [74]
35	Flexibly considering resources during system execution	[15] [30] [31] [38] [47] [52]
36	Task of least resistance: Balancing time, effort, and quality	[21]
37	Synchronously coordinating tasks among workers	[1] [14] [38]
38	Asynchronously coordinating tasks among workers	[15] [42] [75]

work, or being able to submit a higher quality output" [21, pg. 2264]. Allowing workers the agency to pick what they work on can increase performance.

As workers perform tasks that create, update, and evaluate content through a crowd-writing workflow, systems need to coordinate task inputs and outputs among workers. This is not a concern that the solo writer, working alone, needs to worry about. Crowd-writing systems coordinate task inputs and outputs either asynchronously or synchronously. Implementing either coordination method can be challenging because crowd workers are distributed by nature *[24]* and frequently have little to no obligation to one another *[59]*. Our analysis found a wide range of ways that crowd-writing systems coordinate synchronous (e.g., [1, 14, 38]) and asynchronous work (e.g., [15, 42, 75]).

A key challenge involved with coordinating multiple workers is timing, as workers choose when they are available to work and for how long. ⟨P2⟩ noted that, while building a synchronous team, "some workers would enter late, others would leave early." Systems, and therefore designers, must react and monitor those changes; through iteration, ⟨P2⟩ and their research team decided to integrate communication mechanisms (such as chat and annotations on the writing) into their interface to facilitate interactions between asynchronous workers. ⟨P24⟩ argued that "when tasks become complex" they "need more trained people and people who are willing to invest a longer period of time" in comparison to AMT workers.

Some interviewees argued that synchronous coordination is the best method to obtain quality output. ⟨P24⟩ described an extensive comparison of both synchronous and asynchronous monitoring in similar systems. They concluded that synchronous approaches outperform asynchronous methods on accuracy, even stating that "synchronous has to be when you care about getting 100% accuracy," however, "synchronous is also very expensive" ⟨P24⟩. Work in crowdsourced journalism [1] explores actually appointing a worker to synchronously monitor the content. The content curator's role is to constantly monitor the reporters and make sure they produce high quality work. These findings suggest that methods where quality matters may benefit from the implementation investment of a synchronous approach.

Deciding whether to monitor worker coordination asynchronously or synchronously is not a one-size-fits-all decision. In some cases, the design of the workflow or the targeted genre may require one method over the other. For example, synchronous coordination is important for real-time events, such as crowdsourced journalism [1] and real-time Q&A [38]. Monitoring coordinating workers influences how work is done (e.g., crowd memory [38]) and who does it (e.g., roles [1]) for many synchronous systems. However, when a choice exists, one consideration is the cost of synchronous coordination. Drapeau et al. [15] describe their choice of asynchronous infrastructure for MicroTalk as being motivated by cost.

Overall, existing crowd-writing systems concern themselves with practical designs related to Monitor, rather than workers' individual writing styles or progress. This is a reflection of the fact that most systems treat workers collectively, rather than as individuals. While we see these

practical Monitor designs as useful guidelines for future designers, delving deeper into a worker's own Monitor is an appealing idea.

5 DISCUSSION

5.1 What the cognitive process model captures about crowd-writing

The cognitive process model [16] proved to be a robust analytical lens for examining existing research about crowd-writing systems. Our analysis identified 38 design decisions based on a review of 24 papers. The fact that our analysis identified design decisions related to every component of the cognitive process model is noteworthy. Additionally, as the cognitive process model describes the writing process of solo writers, it offers a useful contrast to systems which coordinate crowd workers to perform a sequence of writing tasks. In light of its ease of use for this analysis, we argue that the cognitive process model offers a conceptual map of the crowd-writing system design space, which researchers might use as a guide for future research.

However, the design decisions identified by our analysis are not uniformly distributed across the components of the cognitive process model. Specifically, some writing components have received more attention than others. For instance, our analysis presents six design decisions relevant to Monitor from 14 papers, which may be explained by the substantial interest in crowdsourcing research about how to effectively coordinate processes (e.g., [9, 56, 65]). By comparison, our analysis found fewer systems that design for Goal-setting (3 papers). While these systems have helped to advance research about managing global Goal-setting, future research might focus on local content Goal-setting. As an example of a local content Goal-setting task, a self-guided crowd could traverse a document to identify and correct spelling and grammar errors.

Our analysis also highlights how crowd-writing systems may only rely on a select few key components, rather than design for every aspect of the cognitive process model. For example, Chorus [38] integrates a tight coupling between Memory and Monitor components, which enables its real-time Question and Answer service. As another example, design decisions in CrowdCrit draw primarily from Audience and The Text Produced Thus Far, in order to provide feedback to students [46]. Rather than design for all writing components, these examples demonstrate how designing for a select few may be sufficient to meet a writing objective.

The cognitive process model also offers a way of comparing crowd-writing systems directly. For example, Soylent [7] and CrowdIA [42] use timeouts as methods for monitoring fixed system resources. Although both Storia [30] and the work from Agapie et al. [1] produce narratives (short stories and news articles, respectively), Storia makes use of passive presentation of the growing text, whereas the system in Agapie et al. [1] uses worker roles. Determining how these similarities and differences persist going forward will help us fully enumerate the overall design space. For instance, consistently applying one design may help us assemble libraries of tasks associated with a particular design or component. On the other hand, the perpetuation of similar, but different, designs for the same component may tell us that there is no single best approach, but rather that different styles or genres of writing lend themselves to specific approaches.

Our analysis also identified components of the process model that reflect under-explored areas in existing work. The following components in particular deserve additional consideration through future research: Memory, Organizing, Goal-setting, Translating, and The Writing Assignment. One immediate consideration would be to adapt tools, such as style guides or word banks, to Translating. The dearth of support for workers when crafting specific sentences, paragraphs, etc. is notable, especially given the general focused level of control and design present in other crowdsourcing systems (e.g., *[34]*). At a high level, each of the under-explored components share a procedural aspect, for which potentially further work on workflows or organized tasks are in order. At the

same time, existing work on the lack of sufficiently general workflows (e.g., *Flash Teams* [56], *Flash Organizations* [65]) might suggest that such universal designs for these components may not exist.

5.2 What about crowd-writing is not captured by the cognitive process model

Some aspects of crowd-writing system design are not easily captured by the cognitive process model. The model of a solo writer writing alone and responding to external needs (e.g., from a client, the audience, publishing processes, etc.) does not entirely capture the experience of crowd-writing. Control over the writing process is an important difference. In Flower and Hayes [16], control over the writing rests with the solo writer. In crowd-writing systems, control over the writing is carefully managed by the system, yet rarely in the hands of people within the crowd. The human experience of crowd-writing systems deserves additional attention in our conceptual understanding of this design space. In this section, we focus on a few examples of how control over the process of crowd-writing can play out in the organization of workers.

Many crowd-writing systems coordinate workers performing short, simple, and independent writing tasks in isolation from others within the crowd. In these cases, the worker may have limited knowledge about, and agency over, the parts of the writing process that they can join. Limiting the agency that workers have over the writing process can promote system-level performance. For instance, Soylent's Find-Fix-Verify workflow [7] is robust because it uses three steps to perform a single task, while restricting individuals from participating in both Fix and Verify. Similarly, other systems productively use communication strategies which require workers to discuss or review their work with others (e.g., editors [57], previous workers [15]), rather than allowing workers to determine when they are finished for themselves. On the other hand, providing workers with more agency over the writing process can promote system-level performance by leveraging a worker's existing knowledge [38, 43] and capabilities [21, 30, 38].

Additionally, some crowd-writing systems provide workers with progressive levels of agency over the writing process, but only after they have demonstrated or developed specific capabilities. For example, several existing systems incorporate training activities before workers gain access to a set of writing tasks; CrowdSCIM, for example, teaches workers historical thinking using the SCIM-C scaffold and the paper presents their work as a generalized workflow for teaching expertise [68]. Scaffolds like SCIM-C arose from education where, for example, "progressively *fading* scaffolds, requir[e] the learner to draw on memory to fill new gaps" [46, pg. 484], yet using scaffolds in this manner has not been explored in crowdsourcing or in crowd-writing. Future research should examine the potential for fading scaffolds in task design, especially as crowd workers believe that, "creative work is both beneficial and wanted" [53].

Some crowd-writing systems coordinate people performing writing tasks that involve communication with other people in the crowd. Examples include systems which facilitate worker-worker communication at some stages of the writing process [31] and through specific system components [38, 52]. Some crowd-writing systems are even intended for an entire community of people to share in the writing process (e.g., students, neighbors, reporters [1, 18, 47]). Future research should investigate effective ways of communicating among workers in these novel writing contexts, extending existing research about worker-reviewer communication [57] and worker-client communication [1, 52]. As collaborative writing groups struggle with similar communication barriers [45], this future research may also contribute to a more nuanced understanding of the similarities and differences between collaborative writing and crowd-writing.

Control over the writing process plays out in crowd-writing system design in a variety of ways, e.g., worker agency over the writing process, access to training, and communication among workers, clients, and other participants. These considerations related to worker and process control are reminiscent of organizational behavior challenges faced in business; crowd work is often described

in terms of organizational behavior [32]. Existing crowdsourcing research has applied concepts from the labor organization of guilds in artisan work to consider how crowd workers might collectively reinforce quality standards in production [69]. As an extension of this work, Alkhatib et al. [2, pg. 4608] posited that crowds could "us[e] humans (perhaps other crowd workers) to act as modern *foremen*." Future research might examine opportunities to extend our conceptual map of the crowd-writing system design space, by integrating concepts from organizational behavior with the cognitive process model [16].

5.3 Practical insights from the crowd-writing literature

To place the achievements of crowd-writing systems research in perspective, we can compare current workflows and approaches to an early attempt at crowd-writing. The following is the "iterative writing" approach described by TurKit in 2010:

> "[A]sk one Turker to write a paragraph with some goal. The process then shows the paragraph to another person, and asks them to improve it. The process also has people vote between iterations, so that we eliminate contributions that don't actually improve the paragraph. This process is run for some number of iterations" [44, pg. 63].

As illustrated throughout the findings, the crowd-writing systems developed since TurKit reflect this iterative writing workflow, yet offer numerous contributions to research. Crowd-writing systems research has led to advancements in system and task design, as well as crowd management, by contributing examples of how crowd workers can be supported to perform complex writing assignments.

However, our analysis reveals some common design choices that have the potential to hinder further progress in crowd-writing research. First, the majority of crowd-writing systems included in this review rely on custom-built web platforms (see Table 1). While HCI research partially relies on novelty to explore the bounds of what is possible in system design, a practical concern is that it is hard to replicate and build on custom-built systems that are not shared publicly or that rely on proprietary software. Reusable frameworks for crowd-writing would allow researchers to add, review, and update crowd-writing processes, rather than build these systems from scratch for each study. More broadly, the *Future of Crowd Work* [32] calls for more reusability in crowdsourcing, which has since been explored to a degree by *Flash Teams* [56] and *Flash Organizations* [65].

Second, most systems included in the literature review rely on a single online labor market to recruit crowd workers (see Table 1). Amazon's Mechanical Turk (AMT) has remained the dominant online labor market, but has known risks for both requesters and workers [27, 49]. Several systems in our review coordinate writing efforts with students or community members. These recruitment methods introduce challenges that are different from those related to AMT, such as payment as a potential negative incentive to participation [47] and student achievement as a more important objective than writing [18, 46, 73]. Future research might examine the design of writing-specific labor markets, where workers can gain practice developing their writing skills as sentence-shorteners, grammarians, visual designers, and reporters—among other professions we might invent with crowd-writing systems. The labor economics within a writing-specific market might also be fairly different than typical crowd labor markets, like AMT. Future research might extend existing efforts to simplify the worker recruitment process [5] and to fairly compensate workers for their contributions to writing tasks [17, 70].

Third, papers selected for this literature review include a wide range of protocols for evaluating the performance of crowd-writing systems. What remains is methodological work to develop some consistent standards for evaluating crowd-writing systems, in terms of the performance of their specific processes, as well as a system's overall reliability. Similar to how the research

presented in this paper applied the cognitive process model to compare systems, future research might develop several standard evaluation measures to use as a benchmark for comparing the performance among crowd-writing systems. However, writing quality is not easy to evaluate, often relying on subjective judgements. Future measures for evaluating writing quality may take many forms, such as examining specific writing attributes (e.g., clarity or specificity ⟨P14⟩) or by soliciting field-experts for their opinion (as suggested by ⟨P11⟩).

Finally, this paper has presented a conceptual map of a system design space, by borrowing concepts from the cognitive process model [16] to organize the designs behind a collection of systems that coordinate crowds of people to produce writing. While our analysis points toward future opportunities for crowd-writing research, it is less clear whether there is a community of researchers eager to move this work forward. Few researchers who work on these systems self-identify as "crowd-writing researchers," rather most see themselves as specializing in HCI, machine learning, or other well-established disciplines. A research community is needed to make progress on the practical considerations outlined in this section alone. Our hope is that this paper might serve as a starting point for a conversation about next steps in crowd-writing research, among interested colleagues.

6 LIMITATIONS

The research presented in this paper includes several limitations, including the literature review selection criteria and differences in the intent and style of writing among the selected literature.

The analysis centers on papers published in selected academic venues (CHI, UIST, CSCW, HCOMP), which means that we may have missed a potentially rich series of innovative work that has developed in industry or in other research venues. As the literature search was conducted in 2019, the collection of papers spans nearly a decade of prior work, from 2010 to 2018, but does not include more recent work (e.g., [25]). The literature review also selected systems which generate original content; however, we chose to exclude systems that yield writing as a by-product of another objective, such as image tagging [67], audio transcription [36], and natural language processing-based summarization [28].

Through the analysis process, we recognized that the intent and writing style of an article played into our analysis of the crowd-writing system design. Specifically, the research contributions of some articles clearly extend writing as a crowdsourcing objective (e.g., [29, 43, 52]). Some articles also offer considerable discussion about specific design decisions related to writing tasks (e.g., [21]). However, other papers focus on non-writing contributions, like specific workflow designs [75] or how workers, clients, and communities experience a crowd-writing system [18, 46, 47, 73]. As our categorization of each crowd-writing system depended on the academic writing about the crowd-writing system, articles that were written to showcase the system's writing contributions are somewhat more prominent in the findings.

As our research relies on a literature review and interview study, in which we asked system designers to reflect on their prior work, we cannot speak to what the crowd-writing system design process looks like in action. A researcher's reflection on their process of work might be fairly different from their experience of design work in-situ. As our literature review spans nearly a decade, some of the interviews were also conducted many years after the research concluded. Organizing a workshop at an academic conference, where participants are independently asked to generate several ideas about how to design for specific writing components, while recording their thoughts through the process, might foreground a rich discussion about design practices in this area. Adding aspects of crowdsourcing to curricula may also offer a way to gain a deeper understanding of crowd-writing, as students would provide both a novice perspective and insight into how design patterns can be applied and adapted.

Finally, by conducting a literature review and interviews with crowd-writing system researchers, our analysis prioritized the perspectives of researchers over other participants in crowd-writing—most notably the perspective of crowd workers. The omission of workers' perspectives originates both from our research aim, as well as varied levels of reporting about the worker experience in the selected literature. Recent surveys of crowd workers show that crowd-writing may be desirable work [53], but our analysis does not provide any conclusions or recommendations based on crowd worker experiences. This is a regrettable limitation.

7 CONCLUSION

In this paper, we describe the design of crowdsourcing systems to generate original written text (called *crowd-writing systems*), through an analysis of 24 research papers and interviews with their authors. The analysis applied the cognitive process model of writing [16] as an analytical lens, which we used to group similar system designs together. In total, we identified 38 unique design decisions. Our analysis found that the literature review included design patterns that relate to each of the writing components described in the cognitive process model, but our analysis also highlights design patterns that are specific to crowdsourcing. Our findings suggest future opportunities for research, both in the analysis and design of crowd-writing systems.

8 ACKNOWLEDGEMENTS

We thank the authors we interviewed for their willingness to discuss their experience developing crowd-writing systems. Special thanks to Haym Hirsh for being a champion of this work and a valuable sounding board throughout the research. Additionally, Leah Ajmani and Ian Arawjo provided valuable feedback on early drafts of this paper. Finally, we want to thank the CSCW reviewers for their thoughtful advice.

REFERENCES

[1] Elena Agapie, Jaime Teevan, and Andrés Monroy-Hernández. 2015. Crowdsourcing in the field: A case study using local crowds for event reporting. In *Third AAAI Conference on Human Computation and Crowdsourcing*.

[2] Ali Alkhatib, Michael S Bernstein, and Margaret Levi. 2017. Examining crowd work and gig work through the historical lens of piecework. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 4599–4616.

[3] Nancy Allen, Dianne Atkinson, Meg Morgan, Teresa Moore, and Craig Snow. 1987. What experienced collaborators say about collaborative writing. *Iowa State Journal of Business and Technical Communication* 1, 2 (1987), 70–90.

[4] Amazon. 2020. Amazon Mechanical Turk. https://web.archive.org/web/20200612183816/https://www.mturk.com/.

[5] Daniel W Barowy, Charlie Curtsinger, Emery D Berger, and Andrew McGregor. 2012. Automan: A platform for integrating human-based and digital computation. In *Proceedings of the ACM international conference on Object oriented programming systems languages and applications*. 639–654.

[6] Anne Becker. 2006. A review of writing model research based on cognitive processes. *Revision: History, theory, and practice* (2006), 25–49.

[7] Michael S Bernstein, Greg Little, Robert C Miller, Björn Hartmann, Mark S Ackerman, David R Karger, David Crowell, and Katrina Panovich. 2010. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM, 313–322.

[8] Patrick Biernacki and Dan Waldorf. 1981. Snowball sampling: Problems and techniques of chain referral sampling. *Sociological methods & research* 10, 2 (1981), 141–163.

[9] Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, et al. 2010. VizWiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 333–342.

[10] Minsuk Chang, Léonore V Guillain, Hyeungshik Jung, Vivian M Hare, Juho Kim, and Maneesh Agrawala. 2018. Recipescape: An interactive tool for analyzing cooking instructions at scale. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–12.

[11] Neil Cohn. 2013. Visual narrative structure. *Cognitive science* 37, 3 (2013), 413–452.

[12] Justin Cranshaw, Emad Elwany, Todd Newman, Rafal Kocielnik, Bowen Yu, Sandeep Soni, Jaime Teevan, and Andrés Monroy-Hernández. 2017. Calendar. help: Designing a workflow-based scheduling agent with humans in the loop. In

Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. 2382–2393.
[13] Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G Ipeirotis, and Philippe Cudré-Mauroux. 2015. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In *Proceedings of the 24th international conference on world wide web*. 238–247.
[14] Steven Dow, Anand Kulkarni, Scott Klemmer, and Björn Hartmann. 2012. Shepherding the crowd yields better work. In *Proceedings of the ACM 2012 conference on computer supported cooperative work*. ACM, 1013–1022.
[15] Ryan Drapeau, Lydia B Chilton, Jonathan Bragg, and Daniel S Weld. 2016. Microtalk: Using argumentation to improve crowdsourcing accuracy. In *Fourth AAAI Conference on Human Computation and Crowdsourcing*.
[16] Linda Flower and John R Hayes. 1981. A cognitive process theory of writing. *College composition and communication* 32, 4 (1981), 365–387.
[17] Snehal Gaikwad, Durim Morina, Rohit Nistala, Megha Agarwal, Alison Cossette, Radhika Bhanu, Saiph Savage, Vishwajeet Narwal, Karan Rajpal, Jeff Regino, et al. 2015. Daemo: A self-governed crowdsourcing marketplace. In *Adjunct proceedings of the 28th annual ACM symposium on user interface software & technology*. 101–102.
[18] Elena L Glassman, Aaron Lin, Carrie J Cai, and Robert C Miller. 2016. Learnersourcing personalized hints. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*. ACM, 1626–1636.
[19] Google. 2020. Google Docs. Online. https://web.archive.org/web/20200416081437/https://www.google.com/docs/about/
[20] Nick Greer, Jaime Teevan, and Shamsi T Iqbal. 2016. *An introduction to technological support for writing*. Technical Report. Technical Report. Microsoft Research Tech Report MSR-TR-2016-001.
[21] Nathan Hahn, Joseph Chang, Ji Eun Kim, and Aniket Kittur. 2016. The Knowledge Accelerator: Big picture thinking in small pieces. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2258–2270.
[22] David Hicks, Peter E Doolittle, and E Thomas Ewing. 2004. The SCIM-C strategy: Expert historians, historical inquiry, and multimedia. *Social Education* 68, 3 (2004), 221–226.
[23] Hwajung Hong, Eric Gilbert, Gregory D Abowd, and Rosa I Arriaga. 2015. In-group questions and out-group answers: crowdsourcing daily living advice for individuals with autism. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 777–786.
[24] Jeff Howe. 2019. Crowdsourcing: A Definition. https://web.archive.org/web/20191027010513/https://crowdsourcing.typepad.com/.
[25] Chieh-Yang Huang, Shih-Hong Huang, and Ting-Hao Kenneth Huang. 2020. Heteroglossia: In-Situ Story Ideation with the Crowd. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM.
[26] Ting-Hao Huang, Joseph Chee Chang, and Jeffrey P Bigham. 2018. Evorus: A crowd-powered conversational assistant built to automate itself over time. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.
[27] Lilly C Irani and M Six Silberman. 2013. Turkopticon: Interrupting worker invisibility in amazon mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 611–620.
[28] Youxuan Jiang, Catherine Finegan-Dollak, Jonathan K Kummerfeld, and Walter Lasecki. 2018. Effective crowdsourcing for a new type of summarization task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. 628–633.
[29] Harmanpreet Kaur, Alex C Williams, Anne Loomis Thompson, Walter S Lasecki, Shamsi T Iqbal, and Jaime Teevan. 2018. Creating Better Action Plans for Writing Tasks via Vocabulary-Based Planning. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 86.
[30] Joy Kim and Andres Monroy-Hernandez. 2016. Storia: Summarizing social media content based on narrative theory using crowdsourcing. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*. ACM, 1018–1027.
[31] Joy Kim, Sarah Sterman, Allegra Argent Beal Cohen, and Michael S Bernstein. 2017. Mechanical novel: Crowdsourcing complex work through reflection and revision. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, 233–245.
[32] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. 2013. The future of crowd work. In *Proceedings of the 2013 conference on Computer supported cooperative work*. 1301–1318.
[33] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E Kraut. 2011. Crowdforge: Crowdsourcing complex work. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 43–52.
[34] Nicolas Kokkalis, Thomas Köhn, Carl Pfeiffer, Dima Chornyi, Michael S Bernstein, and Scott R Klemmer. 2013. EmailValet: Managing email overload through private, accountable crowdsourcing. In *Proceedings of the 2013 conference on Computer supported cooperative work*. 1291–1300.
[35] Robert Kraut, Jolene Galegher, Robert Fish, and Barbara Chalfonte. 1992. Task requirements and media choice in collaborative writing. *Human-Computer Interaction* 7, 4 (1992), 375–407.

[36] Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. 2012. Real-time captioning by groups of non-experts. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. 23–34.

[37] Walter S Lasecki, Juho Kim, Nick Rafter, Onkur Sen, Jeffrey P Bigham, and Michael S Bernstein. 2015. Apparition: Crowdsourced user interfaces that come to life as you sketch them. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 1925–1934.

[38] Walter S Lasecki, Rachel Wesley, Jeffrey Nichols, Anand Kulkarni, James F Allen, and Jeffrey P Bigham. 2013. Chorus: a crowd-powered conversational assistant. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. ACM, 151–162.

[39] Edith Law and Luis von Ahn. 2011. Human computation. *Synthesis lectures on artificial intelligence and machine learning* 5, 3 (2011), 1–121.

[40] Mary M Lay and William M Karis. 1991. *Collaborative writing in industry: Investigations in theory and practice*. Baywood Publishing Company.

[41] C Michael Levy and Sarah Ransdell. 2013. *The science of writing: Theories, methods, individual differences and applications*. Routledge.

[42] Tianyi Li, Kurt Luther, and Chris North. 2018. CrowdIA: Solving Mysteries with Crowdsourced Sensemaking. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 105.

[43] Chi-Chin Lin, Yi-Ching Huang, and Jane Yung-jen Hsu. 2014. Crowdsourced explanations for humorous internet memes based on linguistic theories. In *Second AAAI Conference on Human Computation and Crowdsourcing*.

[44] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. 2010. Turkit: human computation algorithms on mechanical turk. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. 57–66.

[45] Paul Benjamin Lowry, Aaron Curtis, and Michelle René Lowry. 2004. Building a taxonomy and nomenclature of collaborative writing to improve interdisciplinary research and practice. *The Journal of Business Communication (1973)* 41, 1 (2004), 66–99.

[46] Kurt Luther, Jari-Lee Tolentino, Wei Wu, Amy Pavel, Brian P Bailey, Maneesh Agrawala, Björn Hartmann, and Steven P Dow. 2015. Structuring, aggregating, and evaluating crowdsourced design critique. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 473–485.

[47] Narges Mahyar, Michael R James, Michelle M Ng, Reginald A Wu, and Steven P Dow. 2018. CommunityCrit: Inviting the Public to Improve and Evaluate Urban Design Ideas through Micro-Activities. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 195.

[48] Thomas W Malone and Kevin Crowston. 1994. The interdisciplinary study of coordination. *ACM Computing Surveys (CSUR)* 26, 1 (1994), 87–119.

[49] Brian McInnis, Dan Cosley, Chaebong Nam, and Gilly Leshed. 2016. Taking a HIT: Designing around rejection, mistrust, risk, and workers' experiences in Amazon Mechanical Turk. In *Proceedings of the 2016 CHI conference on human factors in computing systems*. 2271–2282.

[50] Brian McInnis, Gilly Leshed, and Dan Cosley. 2018. Crafting Policy Discussion Prompts as a Task for Newcomers. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 121.

[51] Microsoft. 2020. Microsoft Office Online. Online. https://web.archive.org/web/20200415074732/https://products.office.com/en-us/free-office-online-for-the-web

[52] Michael Nebeling, Alexandra To, Anhong Guo, Adrian A de Freitas, Jaime Teevan, Steven P Dow, and Jeffrey P Bigham. 2016. WearWrite: Crowd-assisted writing from smartwatches. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 3834–3846.

[53] Jonas Oppenlaender, Kristy Milland, Aku Visuri, Panos Ipeirotis, and Simo Hosio. 2020. Creativity on Paid Crowdsourcing Platforms. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–14.

[54] Peter Pirolli and Stuart Card. 2005. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of international conference on intelligence analysis*, Vol. 5. McLean, VA, USA, 2–4.

[55] Jérémie Rappaz, Michele Catasta, Robert West, and Karl Aberer. 2018. Latent structure in collaboration: the case of Reddit R/place. In *Twelfth International AAAI Conference on Web and Social Media*.

[56] Daniela Retelny, Sébastien Robaszkiewicz, Alexandra To, Walter S Lasecki, Jay Patel, Negar Rahmati, Tulsee Doshi, Melissa Valentine, and Michael S Bernstein. 2014. Expert crowdsourcing with flash teams. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. 75–85.

[57] Niloufar Salehi, Jaime Teevan, Shamsi Iqbal, and Ece Kamar. 2017. Communicating context to the crowd for complex writing tasks. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, 1890–1901.

[58] Matthew J Salganik and Duncan J Watts. 2009. Web-based experiments for the study of collective social dynamics in cultural markets. *Topics in cognitive science* 1, 3 (2009), 439–468.

[59] Gordon B Schmidt and William M Jettinghoff. 2016. Using Amazon Mechanical Turk and other compensated crowdsourcing sites. *Business Horizons* 59, 4 (2016), 391–400.
[60] Ben Shneiderman. 1996. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE symposium on visual languages*. IEEE, 336–343.
[61] Anselm L Strauss. 1987. *Qualitative analysis for social scientists*. Cambridge university press.
[62] Saiganesh Swaminathan, Raymond Fok, Fanglin Chen, Ting-Hao Huang, Irene Lin, Rohan Jadvani, Walter S Lasecki, and Jeffrey P Bigham. 2017. Wearmail: On-the-go access to information in your email with a privacy-preserving human computation workflow. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. 807–815.
[63] The Etherpad Foundation. 2020. Etherpad. https://web.archive.org/web/20200803185137/https://etherpad.org/.
[64] Richard C Thomas. 2012. *Long term human-computer interaction: An exploratory perspective*. Springer Science & Business Media.
[65] Melissa A Valentine, Daniela Retelny, Alexandra To, Negar Rahmati, Tulsee Doshi, and Michael S Bernstein. 2017. Flash organizations: Crowdsourcing complex work by structuring crowds as organizations. In *Proceedings of the 2017 CHI conference on human factors in computing systems*. 3523–3537.
[66] Vasilis Verroios and Michael S Bernstein. 2014. Context trees: Crowdsourcing global understanding from local views. In *Second AAAI Conference on Human Computation and Crowdsourcing*.
[67] Luis Von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 319–326.
[68] Nai-Ching Wang, David Hicks, and Kurt Luther. 2018. Exploring Trade-Offs Between Learning and Productivity in Crowdsourced History. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 178.
[69] Mark E Whiting, Dilrukshi Gamage, Snehalkumar (Neil) S Gaikwad, Aaron Gilbee, Shirish Goyal, Alipta Ballav, Dinesh Majeti, Nalin Chhibber, Angela Richmond-Fuller, Freddie Vargus, et al. 2017. Crowd guilds: Worker-led reputation and feedback on crowdsourcing platforms. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. 1902–1913.
[70] Mark E Whiting, Grant Hugh, and Michael S Bernstein. 2019. Fair Work: Crowd Work Minimum Wage with One Line of Code. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, Vol. 7. 197–206.
[71] Wikimedia Foundation, Inc. 2020. Wikipedia: The free encyclopedia. Online. https://www.wikipedia.org
[72] Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. 2012. Strategies for crowdsourcing social data analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 227–236.
[73] Anbang Xu, Huaming Rao, Steven P Dow, and Brian P Bailey. 2015. A classroom study of using crowd feedback in the iterative design process. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*. ACM, 1637–1648.
[74] Amy X Zhang, Lea Verou, and David Karger. 2017. Wikum: Bridging discussion forums and wikis using recursive summarization. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, 2082–2096.
[75] Haiyi Zhu, Steven P Dow, Robert E Kraut, and Aniket Kittur. 2014. Reviewing versus doing: Learning and performance in crowd assessment. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. ACM, 1445–1455.