Co-destruction Patterns in Crowdsourcing
Formal/Technical Paper

Reihaneh Bidar(✉), Arthur H. M. ter Hofstede, and Renuka Sindhgatta

Queensland University of Technology, Brisbane, Australia
{r.bidar,a.terhofstede,renuka.sr}@qut.edu.au

Abstract. Crowdsourcing has been a successful paradigm in organising a large number of actors to work on specific tasks and contribute to knowledge collectively. However, the openness of such systems allows destructive patterns to form through actors' dynamics. As a result, the collective effort of actors may not achieve the targeted objective due to lower engagement and lower quality contributions. There are varying forms of actor dynamics that can lead to suboptimal outcomes and this paper provides a systematic analysis of these in the form of a collection of patterns, derived from both the literature and from our own experiences with crowdsourcing systems. This collection of so-called co-destruction patterns allows for an-depth analysis of corwdsourcing systems which can benefit a comparative analysis and also assist with improvements of existing systems or the set-up of new ones. A survey reveals that these patterns have been observed in practice and are perceived as worthwhile addressing.

Keywords: Co-destruction · Crowdsourcing · Collaboration · Patterns

1 Introduction

Crowdsourcing systems have become an integral medium for outsourcing tasks to a community of actors [23]. Crowdsourcing refers to the coordination of actors via online collaborative technologies to elicit their knowledge and achieve business goals [47]. The tasks involved typically include problem solving, co-creation of content, evaluating or rating ideas or products [23], and micro-tasking [9]. The integration of actors' contributions in these activities leads to improvement of innovation processes in an organisation [19] on the one hand, and provides job opportunities [32], or intrinsic and extrinsic motivation [19] on the other hand. For example, in the Figure Eight platform, actors provide solutions and receive money in exchange. However, in open source projects such as GitHub, actors contribute code and receive intrinsic rewards in return. In this paper, we focus on content creation and task-oriented crowdsourcing regardless of the type of incentives actors achieve from their contribution.

S. Dustdar et al. (Eds.): CAiSE 2020, LNCS 12127, pp. 54–69, 2020.
https://doi.org/10.1007/978-3-030-49435-3_4

From an economic viewpoint, the total crowdsourcing market is predicted to reach $15–$25 billion in 2020 from $4.8 billion in 2016, increasing the number of people who generate income from crowdsourcing activities [29]. Despite the success of crowdsourcing systems, very little is known about the dark side and failures of crowdsourcing. Malicious workers [32], their low quality contributions, and their dishonesty [48] are examples of challenges that may cause a crowdsourcing task to fail. Collaboration among actors does not necessarily elicit valuable ideas and it is possible that actors behave counterproductively in crowdsourcing initiatives, for example through mockery or by pushing their own agenda [55]. The negative impacts on a crowdsourcing initiative, and their roots in actor behaviour, are referred to as *co-destruction* and form the focus of this paper.

In this paper, a systematic analysis of various forms of co-destruction in the context of crowdsourcing systems is conducted. As co-destruction in this context is still ill-understood, we have chosen a patterns-based approach [2], to be able to identify core recurring phenomena, their affect, their manifestation, and their detection. A patterns-based approach has the advantage of being technology-independent, sufficiently precise though not overly formalised (which would limit the range of interpretations at too early a stage), multi-faceted (allowing one to focus on different aspects of the issue), and extensible. A collection of patterns form a repository of knowledge that can be added to over time, and, in this case, form the foundation of understanding and mitigating co-destruction phenomena in a variety of crowdsourcing systems.

In the rest of the paper, relevant work related to co-destruction and crowdsourcing is discussed in Sect. 2. In Sect. 3 a set of six co-destruction patterns is proposed, each described in some depth. Section 4 analyses the potential manifestation of the co-destruction patterns across a number of crowdsourcing platforms, and Sect. 5 outlines an evaluation where occurrence and perceived importance of the patterns was assessed by users of and contributors to crowdsourcing initiatives. Section 6 discusses the findings and further work.

2 Related Work

The notion of co-destruction has been introduced in the Service Dominant Logic literature as a negative outcome of interactions within a service system [22] and diminished well-being of at least one of the parties involved in the collaboration [41]. Different disciplines have started to explore the dark side of participation in social media [4], and the sharing economy [38]. In a crowdsourcing context, the dark side of crowdsourcing initiatives is an emerging research area (e.g. [55]). These studies identified that technological innovation tools create adverse consequences on society that are worthy of research attention [4]. However, co-destruction resulting from actor collaboration in crowdsourcing systems is a topic that has been overlooked in the literature. It is noteworthy observing that although actors' contributions to crowdsourcing help organisations by

providing diversity in ideas, knowledge, skill, and experience [14], mitigating negative consequences will make the integration of these contributions more effective and limit negative outcomes [55].

The focus of existing studies on co-destruction has mostly been on the reasons why it happens through collaboration and what its consequences are. The distinct forms of co-destruction have been ignored. Vafeas, Hughes, and Hilton [52] found inadequate communication, the absence of trust, power imbalance, inadequate coordination, and inadequate human capital to be the main reasons for failure of contribution integration. Smith [49] found co-destruction to be a failure of the integration of processes which results in unexpected resource loss (i.e., material, social, or energy), perceived misuse, and decline in actor well-being. Gatzweiler et al. [18] found violations of terms and conditions, and questioning of contributions/platform/actor-provided content as forms of deviant behaviour that occurs in crowdsourced ideation contests. Our purpose, however, is to pinpoint patterns that can emerge from actor collaboration and which may result in destructive outcomes whereas the literature has typically emphasised positive outcomes of such collaboration.

3 Co-destruction Patterns

We use a patterns-based approach to characterise various forms of co-destruction in crowdsourcing systems. The use of a pattern-based approach has been suggested as a systematic approach to "identifying, classifying, and documenting the available knowledge", and presenting best practices as solutions to recurring challenges [37]. More specifically, a pattern-based approach is suitable when we are dealing with understanding and characterising a reoccurring problem in a complex domain [16]. For the identification of the various co-destruction patterns we draw upon relevant literature as well as our own knowledge and experience.

Patterns collections may sometimes draw the criticism that they are not "complete". In this regard it should be pointed out that this requires a framework in which completeness can be assessed, which in turn means that we already have a solid understanding of the domain, thus obviating the need for patterns in the first place. Hence pattern collections are more appropriately referred to as comprehensive. Even here though one may argue that patterns need to be able to stand on their own and a pattern collection of valuable patterns is a worthwhile contribution to the field, which can be extended over time.

In this section we present six patterns, each of which we believe encapsulates an important fundamental problem contributing to co-destruction in crowdsourcing systems. For each of these patterns, a description and one or more real-life examples are provided as well as how the occurrence of the pattern may have a negative effect on crowdsourcing outcomes (e.g. quality, timeliness) and participants (e.g. engagement), how it manifests itself, and how it can be detected. The pattern collection provides a repository of knowledge in relation to mitigating or even preventing occurrences of co-destruction in crowdsourcing systems.

3.1 Collusion

Description. Collusion happens when groups of actors make joint decisions contrary to social rules and may result in the colluding actors having an unfair advantage over other interested actors [28]. An alternative definition refers to collusion as a situation where participants violate protocol rules by engaging in activities to gain self-benefit [57]. Collusion in crowdsourcing can result from (i) any form of vote manipulation such as intentionally biasing a rating [28], (ii) endorsing or sinking a product/service/task [3], and (iii) copying information or other people's work [28].

Real-Life Example. In Amazon Mechanical Turk (AMT), a task based on product reviews can be degraded by a malicious group of workers which can result in misleading feedback on or a negative review of the task [26]. Researchers found that Amazon's rating system is subject to collusion and unfair rating [20]. They also found that 35% of Amazon groups (7 out of 20 in the sample log collected by Leskovec et al. [35]) are collusive groups [3]. Collusion in the case of Amazon manifests itself as a "war of reviewers" in which a group of members are regularly corralled to write glowing reviews for themselves to increase their income or to write a negative review about competitors to increase their own reputation [20].

Affect. The occurrence of this pattern may result in strong biases in task outcomes (e.g. final aggregated ratings) which may mislead other actors' perceptions [28] and the "quality of future peer behaviour" [31]. Collusive behaviour results in tricking the system [11] in order to accumulate trust and achieve promotion in the system [10]. Overall, the damaging effect of such behaviour is to negatively influence the quality of outcomes (e.g. the final rating scores of products) and expected behaviour of the system [3].

Manifestation. This pattern manifests itself through different forms of abnormality in crowdsourcing systems such as suspicious reputation system ratings and noise caused by collusive groups. For example, one indicator of a suspicious rate is when a collusive actor gives extreme low or extreme high ratings for the target products they intend to boost or sink [28].

Detection. Collusion indicators have been defined for the detection of collusion for online rating tasks: (i) *Group Rating Value Similarity* helps identify groups of users posting similar ratings for similar products, (ii) *Group Rating Time Similarity* accounts for users promoting or demoting a product in a short time frame, (iii) *Group Rating Spamicity* indicates a high number of ratings posted by a single user on the same product, and (iv) *Group Members Suspiciousness* computes a metric reporting suspicious users based on their ratings as compared to computed credible ratings [3]. Collusion in crowdsourcing platforms such as AMT can be detected by computing and identifying strong inter-rater dependence across tasks especially when they diverge from the mean [28]. Another approach to detecting collusion resulting from actors using collective intelligence

or indulging in group plagiarism considers the actor's ability and label repetitions. The primary assumption is that colluding actors are of lower quality or expertise and produce repeat labels or plagiarised labels [8].

3.2 Bias

Description. Humans have an inclination towards one opinion over another because of previous experiences and perceptions [5]. When this inclination is based on a contributor's subjective judgement, it may create a bias which may lead to unfair decisions toward others [13]. Personal preference can cause bias [54]. For example, bias can result in higher ranked contributors in a crowdsourcing system getting positive rates and feedback because of their reputation and not solely because of their effort and skill in completing a task.

Real-Life Example. In GitHub, a high correlation between reputation (technical and social), geographical location, and pull request acceptance is an example of how bias may manifest itself. Reputation positively influences a developer's decision to accept a pull request [43]. Also, the likelihood of pull-request acceptance rate is found to be 19% higher when the actors (submitter and integrator) are from the same geographical location [43].

Affect. Bias influences an actor's performance and decision making [5,30] and may shift the opinion of actors to an incorrect answer/solution [25]. An example of influence of bias on decision making in Kaggle is that solutions with high public scores are more likely to be selected as final solutions than solutions with high private scores during the submission phase [30]. Thus, the presence of the Bias pattern can be destructive to the functioning of the crowdsourcing platform [30].

Manifestation. The most common forms of the Bias pattern identified by Baeza [5] in the context of crowdsourcing systems are: i) activity bias (or wisdom of a few) "many people do only a little while few people do a lot", and ii) data bias where the content is limited to a few topics. Bias in Wikipedia can be explicit where an article supports a specific point of view. It can also be implicit where the article focuses on one aspect of the topic but omits another equally important aspect [24]. The Bias pattern can manifest itself through properties of tasks in crowdsourcing systems such as AMT, e.g. visual presentation [25].

Detection. Kamar et al. [25] introduced a learning model for automatic detection and correction of task-dependent biases using probabilistic modeling of crowdsourced tasks and actors. The authors address scenarios where actors (or workers) work on consensus tasks. A consensus task is a type of task where the correct answer is unknown to the task owner and the answer is derived via the aggregation of answers provided by actors. Probabilistic graphical models are used to learn and infer the relationships among actors' biases, their answers (or annotations), ground truth task answers (called the labels), and task features. In certain crowdsourcing tasks, actor biases can be detected by 'seeding' a few control questions for which the answers are known into the main task without

letting the workers know which are the control questions. The number of control questions required to detect bias has been another area of research [36]. As goal of the detection mechanism is to provide quality control mechanisms, it does not distinguish biases due to lack of knowledge, actor background, or actor behavior.

3.3 Incompetence

Description. Incompetence refers to an actor's lack of sufficient capability, knowledge, experience, or skill [15,27] and reflects the quality of an actor's work on a task [27,44]. A mismatch between an actor's competence and task requirements leads to poor outcomes. For example, incompetent actors can produce low quality results because of inadequate understanding of task descriptions [17].

Real-Life Example. An example of incompetence concerns a labelling task in AMT, where actors were asked to label pages of digitised books. Although incompetent actors spent considerable time on task, they only provided low accuracy outcomes with low credibility due to their lack of skills, competencies, and their poor understanding of the task. Interestingly, most of the incompetent actors did not return after a not so successful first experience [27].

Affect. Possible consequences of incompetence are lower accuracy in task delivery [27], poor results and unsuccessful completion of the project [15], and reduced overall effectiveness of the crowdsourcing system [17]. In some cases, to maintain their reputation actors participate in tasks which are beyond their skill level, which affects the effectiveness of crowdsourcing systems [17]. The performance of an actor is an indicator of their competency [17].

Manifestation. Incompetence manifests as poor quality of the task being performed [17]. Actors may spend considerable amounts of time on tasks and yet yield poor results.

Detection. Detection mechanisms include identifying actor bias, expertise, and relationship with the task. A Bayesian Classifier Combination model is used to model the relationship of an actors' bias or competence, the output labels provided by them, and the true labels of the tasks [25]. To avoid this pattern, correctly evaluating an actors' actual competence levels is one of several methods to perform quality control in crowdsourcing systems [44].

To reduce the occurrence of this pattern in crowdsourcing systems, using a prototypical task as a pre-test and self-assessments within the pre-selection process is suggested to increase the probability of having more competent actors in task contribution. This method has been evaluated on a real-world sentiment analysis task and an image validation task [17]. The results for the sentiment analysis task revealed over 15% improvement in accuracy and 12% improvement in agreement among actors compared to the traditional performance-based method [17]. The detection of actor incompetence can lead to a competence-weighted approach with more weight given to replies from individuals with higher capabilities than from others in the crowd [44]. However, the difference between the estimated and actual competence of actors is highly related to the success or failure of competence-weighted approaches in crowdsourcing systems [44].

3.4 Vandalism

Description. Vandalism refers to "ill-intentioned contributions" to crowdsourcing tasks which are destructive in terms of the quality of "collective intelligence" [21]. Vandalism is characterised by modifications made by individual actors with bad intentions, initiating spam and producing inappropriate content [1]. For example, malicious edits in Wikis are a common form of vandalism [53]. In open-source projects such as OpenStreetMap (OSM), actors can make changes to the dataset which cause damage to the project [40].

Real-Life Examples. One of the most common examples of vandalism can be seen in Wikipedia. Any deletion, addition, or change of content made "in a deliberate attempt to compromise the integrity of Wikipedia" [39] is a form of vandalism. For instance, among 200,000 edits per day being conducted on average in Wikipedia, 7% are found to be vandalism [1,53]. Vandalism in OSM occurs in the form of active data corruptions. Applicability and reliability of crowd-sourced geodata, as well as the success of the whole project, are heavily affected by such cases of vandalism. During the testing phase in August 2012, the prototype marked around seven million edits as potential vandalism [40].

Affect. The presence of this pattern causes problems such as spending more effort on vandalism mitigation than on task contributions [1,42]. Mitigating vandalism requires time and effort of many people [42]. Furthermore, vandalism causes a crowdsourcing system to be unreliable and makes it hard to produce high-quality content [1]. Additionally, it has been found that first-time contributors have a higher potential to be affected by vandalism [53].

Manifestation. Depending on the nature of the crowdsourcing platform, the way this pattern manifests itself is different. In Wikipedia, vandalism manifests itself as actors participating in malicious blanking/deleting (e.g, removing significant parts of pages or even whole pages), malevolent editing, and the provision of spam links (e.g., use of disruptive, irrelevant, or inappropriate links) [39]. In OSM, this pattern manifests itself in different ways such as randomly deleting existing objects, generating fictional objects, use of automated edits (bots) in the database, and copyright infringements (e.g., use of data from Google Maps) [40].

Detection. The most common approach for detecting vandalism is the use of automated anti-vandalism bots [39] which utilise heuristics based on the number of edits, the size of the edit, whether the editor is anonymous or not, and many other criteria. In Wikipedia, for example, bots have been used to check every edit in real time, to spot vandalism, and revert them if necessary [39]. A similar rule-based vandalism detection method has been applied to detect common types of vandalism in the OSM database [40]. Adler et al. [1] introduced two types of vandalism detection problem: immediate vandalism (i.e., occurring in the most recent revision of an article), and historical vandalism (i.e., occurring in any revision including past ones). A machine learning classifier is used. Features include metadata elements such as the time since the article was last edited, textual features such as the use of uppercase and digit ratio, language features such as the use of biased or bad words, and reputation of the user.

3.5 Domination

Description. Domination refers to uneven power dynamics among actors in crowdsourcing systems. Through domination, a contributor who holds formal decision-making power may deliberately violate privileges and ethical rules of the platform in order to suppress, influence, or modify the contributions of others. For example, domination may present itself through a phenomenon referred to as "wisdom of few" where higher ranked or more popular actors generate most of the content [5].

Real-Life Examples. This pattern occurs in crowdsourcing platforms where there is some organisational structure such as Wikipedia or GitHub. In Wikipedia, dyadic dominance can occur when one user undoes edits of another user or redoes them their own way. Third-party dominance can arise when a third user restores the edits of a user which was undone by another user [33].

In crowdsourcing systems such as OpenStreetMap and Wikipedia, a small group of actors contribute very significantly, while a very large group of actors participate only occasionally [6]. In English Wikipedia for instance, the first version of half the entries was posted by only 0.04% of editors, which means only a relatively small number of actors on Wikipedia are actively contributing [5].

Affect. The affect of domination of small groups over others is that it creates a high risk of "elite capture" or strong demographic bias [6]. Since the dominant actor decides which other actors to hire or reward, the bargaining power balance is in favor of the dominant actor [12]. Therefore, Domination can thrive when a dominant actor possesses a superior position and other actors have limited power [12].

Manifestation. In privilege-based crowdsourcing systems the dynamic power of actors can be characterised as soft power or hard power. An example of soft power is actors receiving a particular label or badge (social or task-oriented) such as Guru, Expert, Deputy, based on their level of contribution to the system. Therefore, other actors rely on their opinion more because of their status. On the other hand, an editor deciding whether a contribution is accepted or not provides an example of hard power. In hard power, a dominant actor has more authority to govern contributions and this may be guided by their preferences. Hierarchies derived from edits in Wikipedia typically had anonymous users occupying the lowest positions in the hierarchy and registered users in the highest positions, confirming the intuition that registering is necessary for acquiring credibility [34].

Detection. A Wikipedia edit network captures three types of interactions: dyadic undo, dyadic redo, and third-party edits. From such a network a pairwise dominance ordering can be derived. The pairwise dominance ordering can be used to derive the actors' hierarchical position as a function of their editing activity [34]. In GitHub, a statistical model is used based on data from the pull requests of projects. The model shows a positive association between pull request acceptance and submitters with a higher number of followers or having a higher status (such as collaborator) [51], thus detecting manifestations of dominance based on actors' social connections.

3.6 Stasis

Description. Stasis refers to the lack of progress in a contributed task due to inaction or adversarial action by contributors. Different forms of the stasis pattern in crowdsourcing are: (i) a situation in which actions are done and undone over a period of time without reaching a resolution; (ii) a situation in which there is a lack of response to an actor's contribution to a task. For instance, a contribution remains in the submitted status because of approval not being granted by another contributor.

Real-Life Example. An example of the stasis pattern in Wikipedia is an "edit war" among editors of articles. In such a conflict, actors express their disagreement toward the opinion of other actors by modifying content that has been authored by them [34] and this process repeats itself ad infinitum.

Affect. There is a relation between the occurrence of stasis and an actor's reputation [33] and stasis may lead to vandalism. For example, actors with a higher reputation initiate most undo-redo cycles, and they are less likely the recipients of such destructive cycles [33]. Such cycles can be a manifestation of domination [34] where actors violate their privileged status to control other actors, either by not responding to their contributions or by refusing them. This behaviour is significant as actor contributions may be lost and this may negatively influence quality [7].

Manifestation. This pattern can manifest itself through lengthy and protracted undo-redo cycles, whose root causes are points of disagreement [34] between actors. These cycles can be found in the logs of crowdsourcing systems and are sometimes also visible to other participants (as in the case of Wikipedia). Another manifestation of stasis is the lack of contribution by actors. In GitHub, examples of lack of contributions are an owner not maintaining a project or abandoning it altogether. A study [45] found that in GitHub only 10% of users can be considered active. Hence, a lot of open source projects die from lack of contribution.

Detection. A controversy measure is computed for each article to detect an edit war in Wikipedia. The controversy measure considers mutual reverts made by the editors (two editors reverting each others' edits). The measure also accounts for the rank or reputation of the editors [50]. In GitHub, a linear regression model is used to identify statistically significant factors that influence the latency of a pull request. Factors such as first response time of reviewers, pull request complexity, and integrator workload are taken into consideration, and predictors among these are identified [56].

4 Comparative Insights

In this section, we analyse each pattern across different crowdsourcing systems to represent their existence and identify mechanisms provided by the system that impede the occurrence of the pattern. Table 1 summaries our analysis.

The ratings can take the form '+' which indicates the existence of a co-destruction pattern in the crowdsourcing system, '+/−' which suggests that some mechanisms exist that prevent the pattern's occurrence, and '−' which indicates that it is unlikely for the pattern to occur in the system.

Wikipedia: Collusion can manifest in Wikipedia when a group of editors can update and edit articles that can overemphasize their point of view. The process of reviewing prevents some forms of collusion (+/−). Bias can occur in the form of 'language bias' and 'reputation bias', though the core policy of Wikipedia is to provide a neutral point of view. By having many editors and reviewers the widespread presence of bias can be mitigated (+/−). Incompetence occurs when uninformed users edit articles. The review mechanism allows for correction of incompetence (+/−). Domination and deference can exist through the linear hierarchy of reviewers (+). The effects of Vandalism are not likely to be long-lasting as there are vandalism detection bots and reviewers who can block such updates (−). Stasis occurs due to constant edit wars on Wikipedia (+).

GitHub: Collusion in GitHub does not exist as a group of contributors define the project, different types of tasks and the task owners (−). Studies show the existence of geographical bias and reputation bias leading to a higher chance of pull request acceptance [43] (+). Incompetence occurs in GitHub as users not familiar with the source code of a project can contribute to that project. The mechanisms of review adopted by users of the system reduce this form of incompetence (+/−). Vandalism does not occur as user edits go through a review process (−). Domination can exist through the linear hierarchy controls that exist for the acceptance of source code pull-request (+). Stasis occurs due to changes made by a developer not being accepted (+).

Waze: Collusion in Waze can occur where a group of actors can simulate non-existent traffic jams [46]. However, the reputation ratings in Waze prevents actors from continually misreporting (+/−). Bias exists with drivers choosing specific roads and routes for reporting (+). Incompetence occurs with new users providing false or incomplete reports, but new users are initially limited with respect to the scope of the updates that they can make and some updates (e.g. road closures) have to be confirmed by other users (+/−). Very few functions require reputation thresholds, thus mitigating the occurrence of domination (−). Vandalism can occur with simulated attacks, but the actors' reputations would be impacted (−). The occurrence of Stasis is difficult as multiple users in the same location need to provide contradictory reports (−).

OSM: Collusion in OSM is low as the system does not provide an unfair advantage in updating the geodata. The correction by local community addresses any updates that are made [40] (+/−). Bias can occur based on the representation of a community of actors in specific geographic locations (+). Incompetence is high as mapping would require the use of the global positioning system or use of aerial imagery picture. In a study on OSM data, 76.3% of the edits by new actors were incorrect. However, in the same study, 63% of the errors were reverted within 24 h and 76.5% within 48 h [40] (+). Domination is low, with actors primarily

updating local information, as there is no hierarchy (−). Vandalism can exist but is corrected by the community and with the help of automated bots (+/−). Statis does not exist, as all updates are immediately available (−).

AMT: In AMT, collusion can occur when a group of actors copy their responses with minor changes, without doing actual work [26]. Reputation mechanisms in AMT prevent actors from colluding continually (+/−). Bias can occur due to actors' perceptions and inclinations. Bias can be detected by considering responses from multiple workers (+/−). Incompetence occurs when actors choose to work on tasks that are not optimally suited to them [17]. The system allows the task creator to specify the skills and competencies required for the task (+/−). Domination does not exist as an actor can choose to work on a task (−). Vandalism is rare as it could impact reputation (−). Stasis does not occur as the task has to be accepted or rejected (−).

The comparative study provides us with insights into mechanisms that prevent or enable specific patterns. For example, the existence of reputation ratings of actors reduces or mitigates occurrences of collusion and vandalism. Incompetence can be high in systems that require specialized skills (e.g. OSM). Domination and Stasis occur when a small group of users claim to have superior positions in the system (reviewers or administrators). Over time, it is hoped that knowledge gained through patterns-based platform analyses can inform the configuration or construction of new platforms.

Table 1. Evaluating co-destruction pattern occurrence in crowdsourcing systems.

Patterns	Wikipedia	GitHub	Waze	OSM	AMT
Collusion	+/−	−	+/−	+/−	+/−
Bias	+/−	+	+	+	+/−
Incompetence	+/−	+/−	+/−	+	+/−
Domination	+	+	−	−	−
Vandalism	−	−	−	+/−	−
Stasis	+	+	−	−	−

5 Evaluation

In order to empirically assess the co-destruction patterns, we focused on two aspects: their *pervasiveness* and their *perceived importance*. To this end, we collected insights from participants that have experience with crowdsourcing initiatives and/or activities, specifically people that have contributed to or initiated crowdsourcing activities. A questionnaire was distributed through social media channels such as LinkedIn Groups and Twitter as well as by email to known contacts. The questionnaire gauged participants' experience with crowdsourcing

and asked for each of the six patterns to what degree they had encountered these patterns and how important they perceived recognition and prevention of these patterns to be (both on a 1–5 Likert scale).

Of the 25 participants who started to answer the questionnaire, a total of 21 participants completed it. 14% of respondents have not contributed to a crowdsourcing platform, but have regularly used GitHub, Wikipedia or StackOverflow. Furthermore, 33% of respondents have edited an article in Wikipedia, 43% contributed source code on GitHub, 48% were owners of a project in GitHub, 38% claimed to have contributed to StackOverflow and 10% to Figure Eight (formerly CrowdFlower). Six out of 21 respondents were involved in initiating a crowdsourcing activity on GitHub, Figure Eight, or a company-specific crowdsourcing platform.

For each of the six patterns there were at least 12 respondents that have encountered that pattern. The most frequently observed patterns were Incompetence, Bias and Stasis (seen by 100%, 90% and 81% of respondents respectively). The Collusion, Vandalism and Domination patterns were least frequently observed (seen by 67%, 67% and 57% of respondents respectively). One respondent believed that the reason behind the lower frequency of some patterns such as Collusion may be that "Community is very much reactive to such events when it happens and terminates its effect quickly". Figure 1 represents the pervasiveness of the co-destruction patterns as observed by the respondents. The recognition and prevention of each of the patterns was perceived as 'Important' or 'Very Important' by at least 55% of the participants (see Fig. 2). Of particular perceived importance were the Incompetence pattern (77% found this 'Important' or 'Very Important'), the Vandalism pattern (77%) and the Stasis pattern (67%). While the sample size of 21 is small, the results are of interest as they provide an indication that these patterns have really been observed across different platforms and that their mitigation is perceived as important by many of the respondents.

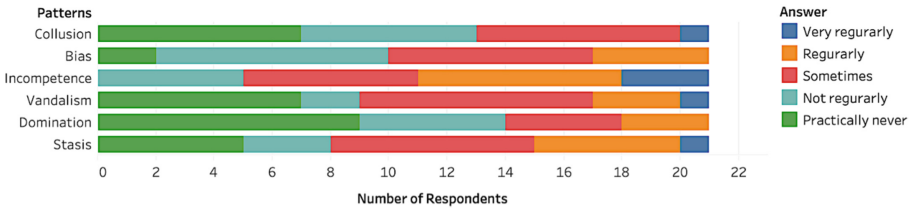

Fig. 1. Pervasiveness of co-destruction patterns

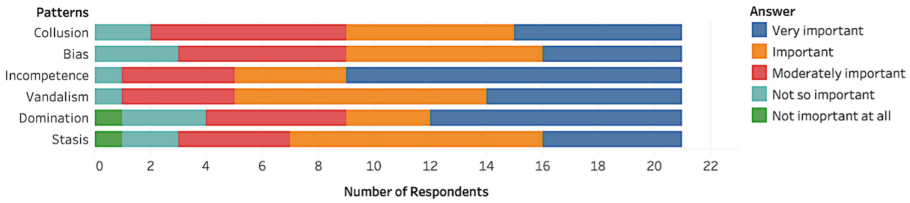

Fig. 2. Perceived importance of co-destruction patterns

6 Conclusion

There are different actor characterics and forms of dynamics in crowdsourcing systems which can lead to suboptimal outcomes and are thus ideally mitigated. In this paper, we provided a systematic analysis of destructive actor behaviour in the form of a collection of patterns, derived from both the literature and from our own experiences with crowdsourcing systems. A survey revealed that these patterns have been observed in practice and are perceived to be of significant importance to be addressed. It was also shown that a patterns-based analysis of some crowdsourcing systems provides a meaningful (comparative) insight. The patterns can be used for benchmarking purposes, for the purpose of selecting, configuring or even developing a crowdsourcing system, and for the development of new detection and mitigation methods (a potential topic for future work). Naturally, the collection of patterns can be extended and refined over time.

References

1. Adler, B.T., de Alfaro, L., Mola-Velasco, S.M., Rosso, P., West, A.G.: Wikipedia vandalism detection: combining natural language, metadata, and reputation features. In: Gelbukh, A. (ed.) CICLing 2011. LNCS, vol. 6609, pp. 277–288. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19437-5_23
2. Alexander, C.: A Pattern Language: Towns, Buildings, Construction. Oxford University Press, Oxford (1977)
3. Allahbakhsh, M., Ignjatovic, A., Benatallah, B., Beheshti, S.-M.-R., Bertino, E., Foo, N.: Collusion detection in online rating systems. In: Ishikawa, Y., Li, J., Wang, W., Zhang, R., Zhang, W. (eds.) APWeb 2013. LNCS, vol. 7808, pp. 196–207. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37401-2_21
4. Baccarella, C.V., et al.: Social media? It's serious! Understanding the dark side of social media. EU Manag. J. **36**(4), 431–438 (2018)
5. Baeza-Yates, R.: Bias on the web. CACM **61**(6), 54–61 (2018)
6. Bott, M., Young, G.: The role of crowdsourcing for better governance in international development. Fletcher J. Hum. Secur. **27**(1), 47–70 (2012)
7. Brandes, U., Lerner, J.: Visual analysis of controversy in user-generated encyclopedias. Inf. Vis. **7**(1), 34–48 (2008)
8. Chen, P., et al.: Collusion-proof result inference in crowdsourcing. J. Comput. Sci. Technol. **33**(2), 351–365 (2018)

9. Chiu, C., Liang, T., Turban, E.: What can crowdsourcing do for decision support? Decis. Support Syst. **65**, 40–49 (2014)
10. Ciccarelli, G., Cigno, R.L.: Collusion in peer-to-peer systems. Comput. Netw. **55**(15), 3517–3532 (2011)
11. Daniel, F., et al.: Quality control in crowdsourcing: a survey of quality attributes, assessment techniques, and assurance actions. ACM Comput. Surv. (CSUR) **51**(1), 7 (2018)
12. Durward, D., et al.: Rags to riches-how signaling behaviour causes a power shift in crowdsourcing markets. In: ECIS (2016)
13. Eickhoff, C.: Cognitive biases in crowdsourcing. In: The Eleventh ACM International Conference on WSDM, pp. 162–170 (2018)
14. Erickson, L., Petrick, I., Trauth, E.: Hanging with the right crowd: matching crowdsourcing need to crowd characteristics (2012)
15. Estellés-Arolas, E., et al.: Towards an integrated crowdsourcing definition. J. Inf. Sci. **38**(2), 189–200 (2012)
16. Fowler, M.: Analysis Patterns: Reusable Object Models. Addison-Wesley Professional, Boston (1997)
17. Gadiraju, U., et al.: Using worker self-assessments for competence-based preselection in crowdsourcing microtasks. TOCHI **24**(4), 30 (2017)
18. Gatzweiler, A., et al.: Dark side or bright light: destructive and constructive deviant content in consumer ideation contests. J. Prod. Innov. Manag. **6**, 772–789 (2017)
19. Gebauer, J.: Crowd resistance-reasons and dynamics of user rebellions against crowdsourcing. In: ISPIM Conference Proceedings, p. 1 (2012)
20. Harmon, A.: Amazon glitch unmasks war of reviewers. The New York Times 14(8) (2004)
21. Harpalani, M., et al.: Language of vandalism: improving Wikipedia vandalism detection via stylometric analysis. In: Proceedings of the 49th Annual Meeting of the ACL: Human Language Technologies, pp. 83–88 (2011)
22. Harris, L., et al.: Not always co-creation: introducing interactional co-destruction of value in service-dominant logic. J. Serv. Mark. **24**, 430–437 (2010)
23. Howe, J.: The rise of crowdsourcing. Wired Mag. **14**(6), 1–4 (2006)
24. Hube, C.: Bias in Wikipedia. In: Proceedings of the 26th International Conference on World Wide Web Companion, pp. 717–721 (2017)
25. Kamar, E., et al.: Identifying and accounting for task-dependent bias in crowdsourcing. In: Third AAAI HCOMP (2015)
26. Kamhoua, G.A., et al.: Approach to detect non-adversarial overlapping collusion in crowdsourcing. In: 2017 IEEE 36th IPCCC, pp. 1–8 (2017)
27. Kazai, G., Kamps, J., Milic-Frayling, N.: Worker types and personality traits in crowdsourcing relevance labels. In: CIKM, pp. 1941–1944 (2011)
28. KhudaBukhsh, A.R., et al.: Detecting non-adversarial collusion in crowdsourcing. In: Second AAAI HCOMP (2014)
29. Kuek, S.C., et al.: The global opportunity in online outsourcing. World Bank (2015)
30. Lee, H.C.B., et al.: Salience bias in crowdsourcing contests. Inf. Syst. Res. **29**(2), 401–418 (2018)
31. Lee, H., Kim, J., Shin, K.: Simplified clique detection for collusion-resistant reputation management scheme in P2P networks'. In: 2010 10th International Symposium on Communications and IT, pp. 273–278. IEEE (2010)
32. Lee, K., et al.: The dark side of micro-task marketplaces: characterizing fiverr and automatically detecting crowdturfing. In: Eighth International AAAI ICWSM (2014)

33. Lerner, J., et al.: The free encyclopedia that anyone can dispute: an analysis of the micro-structural dynamics of positive and negative relations in the production of contentious Wikipedia articles. Soc. Netw. **60**, 11–25 (2018)
34. Lerner, J., Lomi, A.: The third man: hierarchy formation in Wikipedia. Appl. Netw. Sci. **2**(1), 24 (2017). https://doi.org/10.1007/s41109-017-0043-2
35. Leskovec, J., Adamic, L.A., Huberman, B.A.: The dynamics of viral marketing. ACM Trans. Web (TWEB) **1**(1), 5 (2007)
36. Liu, Q., et al.: Scoring workers in crowdsourcing: how many control questions are enough? In: NIPS 2013, pp. 1914–1922 (2013)
37. Lüdeke-Freund, F., Bohnsack, R., Breuer, H., Massa, L.: Research on sustainable business model patterns: status quo, methodological issues, and a research agenda. In: Aagaard, A. (ed.) Sustainable Business Models. PSSBIAFE, pp. 25–60. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-93275-0_2
38. Malhotra, A., Van Alstyne, M.: The dark side of the sharing economy and how to lighten it. CACM **57**(11), 24–27 (2014)
39. Mola-Velasco, S.M.: Wikipedia vandalism detection. In: Proceedings of the 20th International Conference Companion on WWW, pp. 391–396. ACM (2011)
40. Neis, P., Goetz, M., Zipf, A.: Towards automatic vandalism detection in OpenStreetMap. Int. J. Geo-Inf. **1**(3), 315–332 (2012)
41. Plé, L.: Why do we need research on value co-destruction? Journal of Creating Value **3**(2), 162–169 (2017)
42. Potthast, M.: Crowdsourcing a Wikipedia vandalism corpus. In: Proceedings of the 33rd International ACM SIGIR, pp. 789–790. ACM (2010)
43. Rastogi, A.: Do biases related to geographical location influence work-related decisions in GitHub? In: Proceedings of the 38th ICSE Companion, ICSE 2016, pp. 665–667 (2016)
44. Saab, F., et al.: Modelling cognitive bias in crowdsourcing systems. Cogn. Syst. Res. **58**, 1–18 (2019)
45. Sanatinia, A., Noubir, G.: On GitHub's programming languages. arXiv preprint arXiv:1603.00431 (2016)
46. Sanchez, L., Rosas, E., Hidalgo, N.: Crowdsourcing under attack: detecting malicious behaviors in Waze. In: Gal-Oz, N., Lewis, P.R. (eds.) IFIPTM 2018. IAICT, vol. 528, pp. 91–106. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-95276-5_7
47. Saxton, G.D., Oh, O., Kishore, R.: Rules of crowdsourcing: models, issues, and systems of control. IS Manag. **30**(1), 2–20 (2013)
48. Simula, H.: The rise and fall of crowdsourcing? In: 2013 46th Hawaii International Conference on System Sciences, pp. 2783–2791. IEEE (2013)
49. Smith, A.: The value co-destruction process: a customer resource perspective. Eur. J. Mark. **47**(11/12), 1889–1909 (2013)
50. Sumi, R., Yasseri, T.: Edit wars in Wikipedia. In: IEEE PASSAT-SOCIALCOM, pp. 724–727 (2011)
51. Tsay, J., et al.: Influence of social and technical factors for evaluating contribution in GitHub. In: ICSE, pp. 356–366 (2014)
52. Vafeas, M., Hughes, T., Hilton, T.: Antecedents to value diminution: a dyadic perspective. Mark. Theory **16**(4), 469–491 (2016)
53. Viégas, F.B., et al.: Studying cooperation and conflict between authors with history flow visualizations. In: Proceedings of the SIGCHI Conference ACM, pp. 575–582 (2004)
54. Wauthier, F.L., Jordan, M.I.: Bayesian bias mitigation for crowdsourcing. In: Advances in NIPS, pp. 1800–1808 (2011)

55. Wilson, M., Robson, K., Botha, E.: Crowdsourcing in a time of empowered stakeholders: lessons from crowdsourcing campaigns. Bus. Horiz. **60**(2), 247–253 (2017)
56. Yu, Y., et al.: Wait for it: determinants of pull request evaluation latency on GitHub. In: 12th IEEE/ACM MSR, pp. 367–371 (2015)
57. Zou, J., et al.: A proof-of-trust consensus protocol for enhancing accountability in crowdsourcing services. IEEE TSC **12**, 429–445 (2018)