8-10-2021

Collective Attention Allocation for Innovation Productivity in Open-Source Software Projects: A Configurational Perspective

Yanran Liu
Georgia State University

COLLECTIVE ATTENTION ALLOCATION FOR INNOVATION PRODUCTIVITY IN OPEN-SOURCE SOFTWARE PROJECTS: A CONFIGURATIONAL PERSPECTIVE

BY

YANRAN LIU

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree

Of

Doctor of Philosophy

In the Robinson College of Business

Of

Georgia State University

GEORGIA STATE UNIVERSITY
ROBINSON COLLEGE OF BUSINESS
2021

ACCEPTANCE

This dissertation was prepared under the direction of YANRAN LIU's Dissertation Committee. It has been approved and accepted by all members of that committee, and it has been accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Business Administration in the J. Mack Robinson College of Business of Georgia State University.

Richard Phillips, Dean

DISSERTATION COMMITTEE

Dr. Arun Rai (Chair)
Regents' Professor and Howard S. Starks Distinguished Chair
Center for Digital Innovation & Department of Computer Information Systems
J. Mack Robinson College of Business, Georgia State University

Dr. Yu-Kai Lin
Assistant Professor
Center for Digital Innovation & Department of Computer Information Systems
J. Mack Robinson College of Business, Georgia State University

Dr. Likoebe Maruping
Professor
Center for Digital Innovation & Department of Computer Information Systems
J. Mack Robinson College of Business, Georgia State University

Dr. Ling Xue
Associate Professor
Center for Digital Innovation & Department of Computer Information Systems
J. Mack Robinson College of Business, Georgia State University

Dr. Elena Karahanna (External)
Distinguished Research Professor and L. Edmund Rast Professor of Business Management Information
Management Information Systems Department
Terry College of Business, University of Georgia

ABSTRACT

COLLECTIVE ATTENTION ALLOCATION FOR INNOVATION PRODUCTIVITY IN OPEN-SOURCE SOFTWARE PROJECTS: A CONFIGURATIONAL PERSPECTIVE

BY

YANRAN LIU

July 13, 2021

Committee Chair:	Dr. Arun Rai
Major Academic Unit:	Center for Digital Innovation & Department of Computer Information Systems

The proliferation of open-source software (OSS) projects makes it important to understand how to achieve innovation speed and novelty in this context. Although the prior OSS development literature has deciphered how OSS projects can use arm's length mechanisms of digital platforms to self-organize, we lack parsimonious collective-level constructs to illuminate how collectives that achieve favorable innovation outcomes (i.e., high speed, high novelty, and both) organize differently from the others. To address this, we advance a collective attention view by conceptualizing: (1) attention partition, i.e., how collectives differentiate individuals' primary foci of attention and (2) attention augmentation, i.e., how collectives integrate individuals' secondary foci with others' work. We further elaborate each of these constructs based on attention to issues or modules, rendering a novel perspective of collective attention.

Using data of 3,052 release cycles from 363 GitHub machine-learning projects, we conduct fuzzy-set qualitative comparative analyses to investigate how attention partition and attention augmentation by issues and by modules combine to impact innovation productivity of an OSS project in a release cycle. We find that a dual focus (attention partition and attention augmentation) of collective attention with a two-way (issue and module) orientation is a contingency-robust solution to achieve a singular goal of either high speed or high novelty. However, for achieving the dual goal of high speed and high novelty, additional contingencies, including a short release cycle with many developers working on a broad scope of innovative work, are needed. We also find that a one-way dual focus of collective attention can be sufficient to achieve a singular goal as the contingencies vary. Collectively, we contribute a novel dual focus perspective of collective attention for innovation productivity in open-source innovation.

ACKNOWLEDGEMENTS

Along my way to completing this dissertation, many people have helped and supported me. I am happy to have the opportunity to express my gratitude to them.

First and foremost, I want to thank my advisor and dissertation committee chair, Dr. Arun Rai. I sincerely appreciate all his support during my Ph.D. journey. He has invested unmeasurable time and effort in guiding me toward developing an exciting dissertation and research pipeline. During the process, my scholarly capabilities have grown tremendously. I believe that the growth will continue and that his guidance will provide a profound and positive impact on my academic development. Besides, he has helped me to recognize my strengths and weaknesses. He always carefully irrigates my strengths with sincere praises and patiently points out my weaknesses with practical approaches to overcome them. Dots of these persistent efforts connect to develop me as a stronger scholar and person. Moreover, he has made me experience the happiness of working together with someone who I deeply trust. No matter how challenging his comments are, I know that he raises the comments to build me up rather than break me down. No matter how different my thoughts are from his, I know that he will listen closely with an open mind. The collaboration with him is quite enjoyable to me. To conclude, Dr. Rai is a fantastic advisor beyond my imagination!

I also want to thank my dissertation committee members, Dr. Elena Karahanna, Dr. Yu-Kai Lin, Dr. Likoebe Maruping, and Dr. Ling Xue. Their thoughtful and constructive comments have been super helpful in sharpening the problem formulation, construct conceptualization, empirical investigation, and proposition development of my dissertation. Beyond the dissertation, each of them has also given me important guidance during my Ph.D. journey. Dr. Karahanna can always instantly and precisely identify the essence of problems or phenomena. She has inspired me in various aspects of my academic development and self-discovery. Dr. Lin's strong empirical background and rich experience in data collection are amazing. He has given me insightful and actionable advice on how to conceive quality empirical studies and how to collect and manage data appropriately. Dr. Maruping's seminar class exposed me to research concerning collectives (e.g., teams and online communities) at the beginning. His commitment to and knowledge of the domain and theory development is impressive. He has positively influenced me at many stages of my journey. Dr. Xue has generously shared a lot of research experience with me. His suggestions and guidance have been critical to connecting me with different streams of research and have broadened my view to promising future research. Looking back, my growth during the journey is inseparable from their encouragement and support. I very much look forward to future interactions with them.

In addition to my committee, many other scholars have supported, inspired, and encouraged me along my journey. Many thanks to Dr. Lars Mathiassen, Dr. Anandhi Bharadwaj, Dr. Mark Keil, and Dr. Edward Rigdon. I have learned so much from their classes regarding theories and methods. Their tastes for quality research and commitment to academic excellence have inspired me on how to make good decisions in my scholarly roles as a researcher, a reviewer, and an instructor. Their support and encouragement at different stages of my Ph.D. journey have motivated me to move forward confidently and calmly. I would also like to thank my mentors at the ICIS 2020 Doctoral Consortium, i.e., Dr. Sirkka Jarvenpaa and Dr. Chee-Wee Tan, for providing constructive feedback on my dissertation ideas. The discussions with them were enjoyable and inspiring. Also, I want to thank Professor Jing Zhao, Dr. Ning Nan, and Dr. Haixiang Guo. Without their support and encouragement, I would not discover my strong intellectual curiosity and passion for research before joining the Ph.D. program. I appreciate the opportunities to work with them on research projects when I was an undergraduate and a master student. My sincere thanks to other professors, teachers, and mentors who have helped me as I grow up. They have empowered me to achieve my potential and dreams.

Surely, without my family's and my friends' support, I could never reach where I am today. My parents, Guozhen Liu and Guiyin Li, have given me their deepest love. They have unconditionally supported whatever I wanted to do. They made me an appreciative and optimistic person. I love you, Dad and Mom! Turning to my friends, they have shared with me my ups and downs. I truly appreciate their accompany along the journey. To my CEPRIN/CDIN friends, Liwei Chen, Zhitao Yin, Wei Zhang, Weifang Wu, Peiwei Li, Jessica Pye, Vitali Mindel, and Chaitanya Sambhara, thank you very much for making our center a pleasant place to work and grow. I have learned so much from each of you. Your care and help have greatly supported me. To Zhanfei Lei, Xinying Liu, Hengqi Tian, Yukun Yang, Yuting Wang, Fengyuan Liu, Jing Tian, Jinsoo Yeo, Junyoung Park, and Gefei Wuwang, you have contributed so much to my mental and physical health and my happiness. It is difficult to imagine what my Ph.D. life would be like without you. To Pengtao Li and Xuedan Jiang, thank you for always being with me over the 10^+ and even 20^+ years. I am so lucky to have you during my life journey. To other friends who have accompanied me, supported me, and inspired me during my life journey, thank you so much for making me a better person.

Lastly, I would like to thank myself for keeping the passion for research. Getting into something is easy; keeping the passion for it is not. My passion for research has motivated me to persistently pursue excellence with sincere efforts when working on this dissertation. In the future, Yanran, I hope you can keep the passion and sincerity with you. Be a person who has dreams in heart and light in eyes.

TABLE OF CONTENTS

1. INTRODUCTION

This chapter introduces the motivation and research approach of this dissertation. The elaboration of our motivation (see Section 1.1) leads to the formulation of two research questions: (i) how does collective attention allocation impact innovation productivity of an open-source software project in a release cycle? (ii) How does the impact of collective attention allocation on innovation productivity vary with contingencies related to the scope of innovative work, the availability of development resources, and the temporal aspects of a focal release cycle?

We use a three-step approach to answer the research questions: (i) conceptualizing collective attention constructs, (ii) empirically investigating how the collective-attention constructs combine to impact innovation productivity, and (iii) deriving propositions relating collective attention configurations to innovation productivity (see Section 1.2).

1.1 MOTIVATION

With the advance of software development platforms, such as GitHub, developing software based on an innovation agenda publicly shared on the platform has become a central phenomenon. We define such platform-based software development as open-source software development. The publicly shared innovation agenda defines the boundary of an open-source software (OSS) project; it includes two basic elements: (i) a shared codebase and (ii) a list of open or closed issues within the current shared codebase as well as actions to address the issues. An issue can be a bug report or a feature request that potentially leads to a software update such as a patch or a new feature. Developers in the large-scale community of the platform who work on the same publicly shared innovation agenda constitute a community-based collective for the OSS project. As developers take actions to propose and address issues within the shared codebase, software updates are generated, and the software development proceeds. We formulate our research question concerning open-source software development in three steps.

First, we follow an allocation logic to study determinants of speed and novelty of software production. Prior literature on open-source software development has well studied what motivates developers to (continuously) participate in open-source software development (e.g., Bagozzi and Dholakia 2006; Maruping et al. 2019; Zhang et al. 2013; Ho and Rai 2017; Von Krogh et al. 2012; Oh and Jeon 2007; Shah 2006; Roberts et al. 2006; Ren et al. 2016). However, the research inquiry on how to effectively allocate available development resources from involved developers is understudied. Following Ahuja et al.'s (2008) framework, we argue that the former follows an incentive logic and studies determinants of innovation efforts, whereas the latter follows an allocation logic and studies determinants of innovation outputs. Specific to open-source software development, innovation effort refers to development effort (e.g., code changes or issue reports) for an OSS project, whereas innovation output refers to software updates. As software development is a process of trial and error, development efforts are not equivalent to realized software outputs.

We draw on studies in the allocation logic and seek to decipher determinants of innovation speed and novelty in the context of open-source software development. *Innovation speed* refers to the number of software updates an OSS project produces in a fixed time (e.g., per day or week), whereas *innovation novelty* refers to the extent to which the produced software updates demonstrate a breakthrough development. The two innovation productivity measures yield three plausible innovation productivity goals for OSS projects: high speed, high novelty, and both high speed and high novelty. OSS projects that generate software updates with high speed can gain market opportunities (Dong et al. 2019), especially in emergent technology spaces like machine learning. As the software gets mature or obsolete, generating few major or breakthrough updates (i.e., high novelty) can be more important. Although pursuing both speed and novelty can be challenging, OSS projects that achieve the dual goal can gain more advantages. We attempt to decipher how to effectively allocate available development resources to achieve these three innovation productivity goals in a given release cycle of an OSS project.

Second, we seek to decipher collective-level determinants for innovation productivity. Prior literature that concerns how to effectively allocate resources from developers in a community has revealed platform-based technical functionalities and associated practices that enable OSS projects to do so (e.g., Lindberg et al. 2016; Howison and Crowston 2014; Dahlander and O'Mahony 2011). The revealed mechanisms are at the individual-level and contribute insights on how the community-based collective for OSS projects can self-organize to innovate without hierarchical authority or significant managerial force. Building upon these insights, we seek to develop parsimonious collective-level constructs to decipher how collectives can organize to achieve each of the three innovation productivity goals.

As to prior innovation literature (e.g., Kessler and Chakrabarti 1996; Schubert and Tavassoli 2020), although scholars have developed elegant collective-level constructs that depict determinants of innovation speed and novelty in the traditional organizational context of a firm or a formal team, the constructs cannot be directly generalized to the novel context of open-source software development. The reason is that distinctive contextual aspects of open-source software development challenge the underlying assumptions of the theories. For instance, as Nambisan et al. (2017) have identified, a key assumption in theories on innovation management is that innovation is a well-bounded phenomenon focused on fixed products. However, in the context of open-source software development, the innovation space is unbounded. As any developer from the community can propose and work on issues within the codebase, it is difficult to predict how a software product forms or evolves. Consequently, we cannot assume that the conceptualization of the constructs from the traditional organizational context is valid, nor can we assume that the same determinants of innovation productivity hold, in the open-source software development.

Thus, based on prior attention literature in a range of fields, we propose a collective attention view that accommodates the distinctive contextual aspects of open-source software development (see an elaboration of the theoretical view in Chapter 2). In the collective attention view, attention is selection for actions. The distribution pattern of granular actions conducted by individuals in a collective reflects how the collective allocates available development resources from developers in a community. By characterizing the distribution pattern of granular actions, we develop collective-attention constructs (i.e., attention partition and attention augmentation by issues and by modules) that contrast different collectives' allocation of available development resources. These constructs enable us to discern two organizing modes (a division mode and an integration mode) and examine how they affect innovation productivity.

Third, we focus on a dilemma about the scarcity of collective attention for innovative work. This decision is motivated by a ubiquitous phenomenon that community-based collectives for open-source software development consist of a large periphery of developers who propose issues but a small core of developers who take actions to solve the issues (Lee and Cole 2003; Setia et al. 2012; Kuk 2006; Kalliamvakou et al. 2016). A popular open-source software (OSS) project (defined as a repository in GitHub's terminology) can have hundreds or even thousands of open issues to be solved, but only several or dozens of developers who take innovative actions to solve them. For instance, OpenCV/OpenCV, an open-source computer vision project tracked by 41.7 thousand users, had about 1,700 open issues in January 2020. However, only 32 developers took innovative actions (or made commits in GitHub's terminology) to solve issues that month. The limited available development resources and the high demand for innovative work necessitates the problem of collective attention allocation.

In developing our conceptualization of collective attention for innovative work, we specify the focal granular actions that underlies collective attention. Specifically, the observable granular actions that we focus on to conceptualize collective-attention constructs are innovative actions, which introduce changes into the product (i.e., software in the context of open-source software development). These innovative actions are in contrast to supportive actions, which generate and interpret ideas about possible directions pertaining to product change. The underlying selection problem of collective attention allocation is to map limited available development resources from developers in the community to the high demand for innovative work. Community-based collectives may differ in their selections. We conjecture that the differences in selections explain whether they achieve innovation productivity goals regarding speed and novelty.

To conclude, our outcome of interest is innovation productivity, particularly speed and novelty, in the open-source software development. We seek to understand how collective attention allocation affects innovation productivity. Since OSS projects' innovation productivity goals can vary across release cycles, and their collective attention allocation can also vary across release cycles, we regard a release cycle of an OSS project as our unit of analysis. Besides, as the effect of collective attention on innovation productivity can vary with intertwined contingencies related to the scope of innovative work, the availability of development resources, and the temporal aspects (i.e., project maturity and length of release cycle) of a focal release cycle, we also consider the combinational effect of collective attention and contingencies. We formulate our research questions as follows.

Research questions*: How does collective attention allocation impact innovation productivity of an open-source software project in a release cycle? How does the impact of collective attention allocation on innovation productivity vary with contingencies related to the scope of innovative work, the availability of development resources, and the temporal aspects (project maturity and length) of a focal release cycle?*

1.2 RESEARCH APPROACH

We break our approach to answer the formulated research questions into achieving three research objectives: (i) conceptualizing collective attention constructs, (ii) empirically investigating how the collective-attention constructs combine to impact innovation productivity, and (iii) deriving propositions relating collective attention to innovation productivity. In the following subsections, we illuminate the distinctive aspects of the research inquiry and elaborate how we leverage them in attaining the research objectives.

1.2.1 Conceptualizing Collective-Attention Constructs

Distinctive contextual aspects*: Unbounded innovation space, fluid innovation agency, and unsupervised resource allocation in the context of open-source software development*. Conceptualizing collective-attention constructs is to construct a collective attention view that accommodates three distinctive contextual aspects of open-source software development. These three distinctive contextual aspects are unbounded innovation space, fluid innovation agency, and unsupervised allocation of available development resources from developers in the community (hereafter, unsupervised resource allocation).

First, *unbounded innovation space* refers to the continuous influx of issues into the publicly shared innovation agenda of an OSS project. The openness of innovation agenda makes it possible for the large-scale developers who reside on the platform to propose issues at any time. For instance, a popular OSS project on GitHub.com can receive thousands of new issues per year. Consider TensorFlow/TensorFlow, a popular OSS project on deep learning. It received 22,494 issues from November 2015 to January 2020 and is still active with more than 300 new issues proposed per month. The continuous influx of issues forms an unbounded innovation space for an OSS project. Innovation in this context is like navigating through a labyrinth that is quickly expanding, making it cognitively demanding. Without selection, the enormous number of issues can easily deplete the available development resources. The unbounded innovation space also makes it difficult to study a community-based collective's selection (or attention allocation) because innovative work that is requiring attention is changing.

Second, *fluid innovation agency* refers to the unpredictable dynamics of available development resources. Developers are free to enter and leave any OSS project. Consequently, developers' work for the project, even the core developer's work for the project, is intermittent. For instance, Figure 1.1 illustrates the distributions of innovative work from the top ten contributors to the OSS project, OpenCV/OpenCV on GitHub. The highly rugged and erratic distributions of their work reflect the unpredictable dynamics of development resources available for this project. The fluid innovation agency surfaces the limitation of available resources in a given period and increases the difficulty of studying a community-based collective's selection because available development resources to be mapped to the innovative work is also changing.

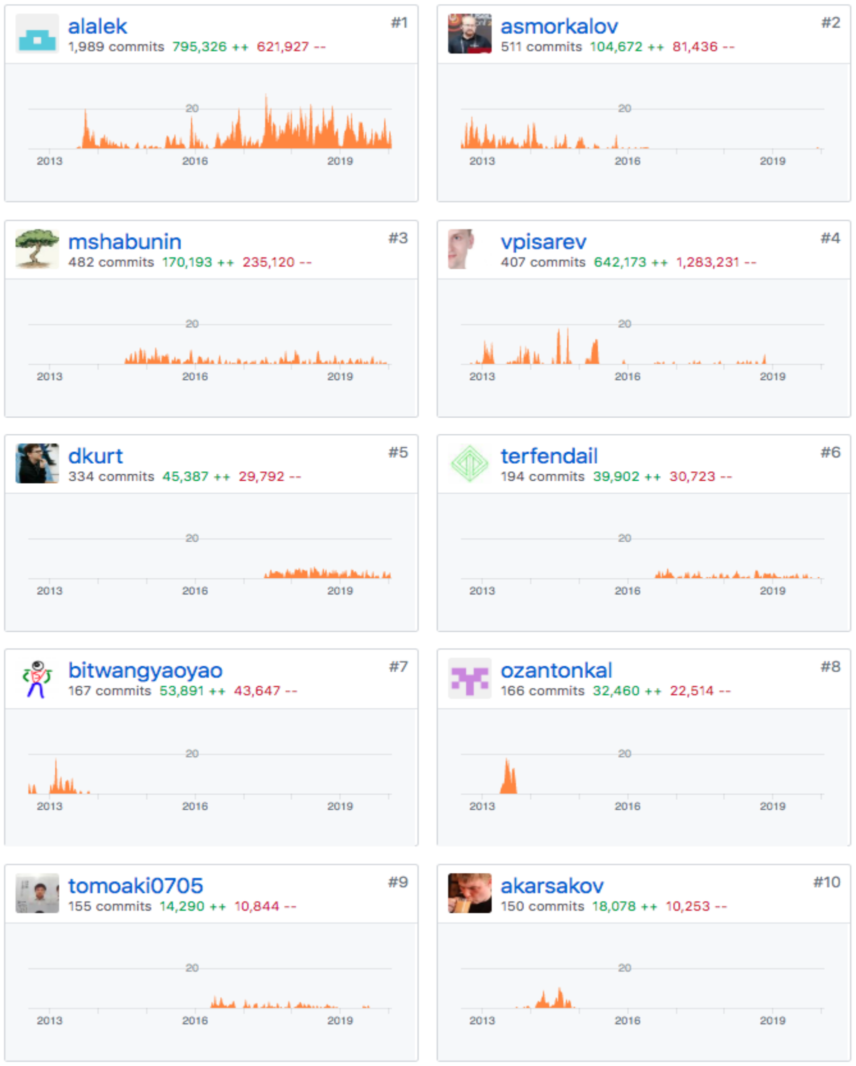


Figure 1.1 An Illustration of the Intermittent Availability of Development Resources from Core Contributors of an Open-Source Software Project

Third, *unsupervised resource allocation* refers to the self-organized nature of this community-based collective innovation. Developers in a community-based collective make their own decisions on where to allocate their limited available resources (e.g., to resolving which issue or modifying which module). What influences their allocation decisions are pieces of code changes and cues (e.g., labels and code reviews) generated by innovative and supportive actions of developers in the community. Unlike managerial commands in the team-based or organization-based collective innovation, the pieces of code changes or cues do not force developers' allocation choice or significantly change their incentives on resource allocation. They nudge developers' thoughts on the choice architecture of behavior. This unsupervised mode of resource allocation makes it difficult to approach the essence of a community-based collective's selection because the forces that guide the formation of the realized mapping route between resources and candidate work are subtle, and the sources of these forces are heterogeneous.

Insights from prior literature*: Observing the pattern of granular actions*. Two critical insights from the prior literature on open-source software development inspire us on how to accommodate these distinctive contextual aspects. First, this literature tends to observe innovation at a much more granular level, i.e., actions that incrementally advance the innovation progress, compared with the innovation literature, which observes innovation at the product architecture, innovation process, individual, project, or organization levels. By zooming into the actions, one will find that for each action, the innovation space (i.e., the target issue of the action) is bounded, and the innovation agency (i.e., the developer who conducts the action) is predefined. For the community-based collective which works on an OSS project, each action represents a piece of the puzzle for the collective's realized mapping route between its available development resources and the work demanding the resources. Thus, the collection of all actions for the project identifies the whole puzzle of the collective's realized mapping route. Observing actions by the collective can be an effective way to accommodate the unbounded-innovation-space and the fluid-innovation-agency aspects of the selection problem.

Second, studies on organizing rationales of open-source software development indicate that although individual developers' actions are unsupervised during innovation in OSS projects, their behaviors are not completely random. There are some general behavioral rules underlying individual developers' actions. For instance, Howison and Crowston (2014) find that most issues of OSS projects are solved by only one individual; issues appearing too large for one individual to solve are more likely to be deferred until they are easier to be solved rather than being undertaken through structured teamwork. Lindberg et al. (2016) indicate that routines exist in OSS projects' coordinative work of mapping individuals to tasks of issue solving. Given the behavioral rules, we conjecture that collective-level patterns should exist in the whole puzzle of actions by individual developers in the same community-based collective. Accordingly, it is meaningful and useful to conceptualize constructs that characterize these patterns to accommodate the unsupervised-resource-allocation aspect of the selection problem.

Theoretical approach to leveraging the distinctive contextual aspects*: Proposing a theoretical view based on a behavioral perspective of collective attention*. The two theoretical insights from prior open-source software development literature motivate us to adopt a behavioral perspective of collective attention and propose a collective attention view based on prior attention literature.

The collective attention view observes actions conducted by the community-based collective that works on an OSS project, and it uses the distribution pattern of the actions to depict how the collective allocates its limited available development resources. Each action refers to an episode of work that can be identified by the issue that the action works on, the developer who conducts the action, the module that the action produces, and the timestamp when the action is pushed to the publicly shared innovation agenda of the project. The identification conditions of an action anchor the actor's attention (i.e., selection for action) at a certain time point. For instance, developer i attends to issue j, if at the specific time point, the developer takes action to solve issue j instead of the many other issues in the publicly shared innovation agenda.

Along this logic, a collection of all actions pushed to the agenda during a given time period anchors the collective's attention in that period. Collective attention refers to the distribution pattern of a collective's actions in the behavioral space of candidate work (e.g., innovative work like solving issues) on the publicly shared innovation agenda that the collective works on. Constructs under the umbrella of collective attention characterize the distribution pattern from distinct aspects.

This collective attention view is different from the attention view in prior management literature in two ways. First, prior management literature has contextualized attention in organizations (e.g., Ocasio 2011; Ocasio 1997; Vuori and Huy 2016; Li et al. 2013; Haas et al. 2015; Weick and Sutcliffe 2006). Drawn on the cognitive psychology and neuroscience literature on attention (e.g., Chapter 4, Sternberg and Sternberg 2016; Petersen and Posner 2012; Pashler et al. 2001), this stream of literature tends to conceptualize attention as specific selection processes or mechanisms embedded in a collective and focuses only on the "brain" (i.e., decision-makers or knowledge providers) of the collective. However, based on prior philosophy literature that has questioned assumptions on the nature of attention in the cognitive psychology and neuroscience literature (e.g., Wu 2014; Wu 2011; Mole 2011), we conceptualize attention as the collective-level order underlying a collective's selection for actions and focus on all individuals of the collective.

Second, prior management literature stays at the cognition layer and assumes that cognitive limitations necessitate attention. However, this dissertation moves to the behavior layer and assumes that the agency (i.e., concrete constraints from action) necessitates attention. Agency includes not only cognitive limitations but also contextual and environmental conditions that constrain the simultaneous execution of all possible actions to avoid behavioral chaos (Neumann 1987). The agency assumption allies tighter with the OSS development context than the cognition assumption. Consider the fact that most developers do not regularly work on an OSS project. Limited work time is probably a more realistic constraint than limited cognition that necessitates attention.

This collective attention view is also different from prior literature that has studied collective attention at the behavior layer. This stream of prior literature conceptualizes collective attention in the context of social media or news media and observes individuals' selection on content consumption (e.g., Lehmann et al. 2012; Wu and Huberman 2007; Sasahara et al. 2012; Mocanu et al. 2015). Individuals of a collective in such contexts lack a common goal and the interdependency among their actions is low. In contrast, individuals in a collective who work on an OSS project share an innovation agenda and their actions are highly interdependent.

To conclude, inspired by prior open-source software development literature, we accommodate the distinctive contextual aspects of open-source software development by proposing a novel collective attention view. This collective attention view conceptualizes collective attention as a collective's selection for actions and advises to develop constructs by characterizing the distribution pattern of the collective's actions in the behavioral space of candidate work from distinct aspects.

1.2.2 Empirically Investigating How Collective-Attention Constructs Combine to Impact Innovation Productivity

The next step in the research process is to empirically investigate the relationship between the collective-attention constructs and innovation productivity regarding speed and novelty. We will develop theoretical propositions based on the empirical results. There are two distinctive aspects of the empirical investigation.

Distinctive form of relationship among constructs: *How the collective-attention constructs combine to elicit innovation productivity outcomes*. The first distinctive aspect is that we need a holistic approach to understand how collective-attention constructs combine to impact innovation productivity of an OSS project in a release cycle. We have four collective-attention constructs, i.e., attention partition and attention augmentation by issues and by modules. If we use an econometrics approach to investigate their combinational effect on innovation productivity, we need to create a long list of interaction terms, and the interpretation of the results can be extremely difficult. Not to mention that we also want to consider the combinational effect of these constructs and contingencies (related to the scope of innovative work, the availability of development resources, and the temporal aspects of a focal release cycle). While understanding the interaction terms can be useful, they can get complicated. We need to also move beyond the interaction terms to decipher how the elements combine in nuanced manners (e.g., redundancy of elements in a combination) to affect the outcome.

Approach to decipher the effect of combinations: *Adopting a configurational perspective*. We adopt a configurational perspective (advocated by El Sawy et al. 2010; Fiss 2011; Park and Mithas 2020) to investigate how the collective-attention constructs (and the intertwined contingencies) combine to impact collective innovation productivity. In the configurational perspective, causal conditions are regarded to impact the outcome as complex configurations of characteristics, rather than a list of independent factors. It approaches the nature of collective attention in the context of open-source software development. Constructs under the umbrella of collective attention characterize the collective-level order (or pattern) of the same distribution of innovative actions from distinct aspects. As the distribution is emergent from individual-level innovative actions and the allocation of these actions is unsupervised, researchers need to understand the actions simultaneously to derive characteristics of the pattern. If the actions cannot be understood separately, constructs that characterize their pattern should not be understood in isolation. Thus, investigating the effect of recurring clusters of collective-attention characteristics or configurations of the constructs would be more relevant than investigating the net effects of each construct.

Besides, the configurational perspective has advantages in deciphering the potential complex causality in forms of equifinality, multifaceted causality, as well as causal asymmetry. In terms of equifinality (or the presence of multiple ways to success), we expect that distinct combinations of collective attention constructs can yield similar outcomes, e.g., “high-speed”, “high-novelty”, or “both high-speed and high-novelty” collective innovation. In terms of multifaceted causality, we expect that the four constructs can suppress, substitute, or complement each other’s effect. For instance, issue-oriented attention augmentation and module-oriented attention augmentation (or attention partition) may elicit similar outcomes, i.e., being substitutive. In terms of causal asymmetric, we expect that configurations for a favorable innovation productivity outcome (i.e., “high speed”, “high novelty”, or “both high speed and high novelty”) may not be simply a mirror image of configurations for its negation (i.e., “low speed”, “low novelty”, or “low speed or low novelty”).

Requirement on observability of granular actions for collective work*: Observing granular innovative actions for open-source software development during a release cycle*. Another distinctive aspect of our empirical investigation is that the collective attention view we employ requires granular and comprehensive observability of software development activities. As the collective attention view regards all developers as stakeholders in a collective’s attention allocation, we need to observe innovative actions conducted by every developer. Besides, for each innovative action, we need to identify not only the developer but also the associated issue(s) and module(s) as we are interested in collective attention allocation with respect to both issues and modules.

***Approach to address the requirement**: Relying on GitHub APIs*. Modern platforms for software development enable the comprehensive observability of software development activities. We select GitHub platform as our empirical setting for studying open-source development. For an OSS project on GitHub, all its software development activities over history are observable; how the activities situate in the publicly shared innovation agenda of the project is also observable (Kalliamvakou et al. 2016). GitHub REST API (application programming interface) and GitHub GraphQL API enable us to access the necessary information: innovative action (i.e., commit in GitHub's terminology) by a developer for an issue and for a module.

1.2.3 Deriving Propositions Relating Collective Attention to Innovation Productivity

***Needs to achieve powerful parsimony in theorizing**: Interpreting complex configurational results*. The distinctive aspect of our proposition development is to interpret configurational results. We will need to interpret six sets of major configurational results: the association of three innovation productivity goals (i.e., high speed, high novelty, and both high speed and high novelty) with two groups of causes: (i) configurations of the collective-attention constructs and (ii) a more elaborate set of configurations involving the collective-attention constructs and the contingencies.

Making sense of each set of configurational results can be challenging as configurational results reveal “recipes” (or combinations) of variables sufficient for achieving a specific outcome. As such, each proposition that links the configuration to the outcome can involve several causes. Accordingly, researchers need to reason why the elements together produce the outcome, not just why each condition benefits or constrains the outcome. If there is equifinality, researchers also need to contrast the equifinal configurations and assure that the interpretations of those equifinal configurations are logically consistent.

Approach to powerful parsimony in theorizing*: Starting from the comprehensive configuration and proceeding to illuminate the redundancy of included elements*. Our overall interpretation strategy is that we start from the collective-attention configuration that is sufficient to elicit all three innovation productivity goals. We find that the comprehensive configuration of collective attention (i.e., attention partition and attention augmentation by both issues and modules) is sufficient to achieve a singular goal (i.e., either high speed or high novelty). When combined with appropriate contingencies, this configuration can also achieve the dual goal of both high speed and high novelty. Then, keeping all other elements the same, if the absence of an element in the configuration elicits the same outcome, we interpret this element as redundant. It means that the presence or absence of this element does not make a difference. Adopting this strategy, we further interpret equifinal configurations of collective attention (and contingencies) by reasoning why some elements become redundant for achieving a certain goal under certain contingencies.

Finally, in the following chapters, we will explain:

- How we develop the collective attention view and our theoretical framework, which includes collective-attention constructs, intertwined contingencies, and innovation productivity regarding speed and novelty (see Chapter 2),

- How we design an empirical study to investigate which configurations of collective attention (and intertwined contingencies) elicit the attainment of three innovation productivity goals, i.e., high speed, high novelty, and both (Chapter 3),
- The analytical results of our empirical investigation and their robustness (Chapter 4), and
- How we interpret the analytical results and develop theoretical propositions (Chapter 5).

2. THEORETICAL FRAMEWORK

In this chapter, we elaborate the theoretical principles of the collective attention view. These principles indicate (i) what aspects of the phenomenon are in the foreground and background when the collective attention view is applied and (ii) why this view is a useful perspective for the open-source software development phenomenon. We proceed to apply the collective attention view to conceptualize four constructs (i.e., attention partition and attention augmentation by issues and by modules) that characterize collective attention allocation. Finally, we develop a theoretical framework that takes a configurational perspective to investigate the relationship between how the collective attention constructs combine to affect innovation productivity (regarding speed and novelty) of an open-source software (OSS) project in a release cycle and how the configurations of collective attention for different productivity goals vary with intertwined contingencies.

2.1 THEORETICAL PRINCIPLES OF THE COLLECTIVE ATTENTION VIEW

We start with elaborating the theoretical principles to answer two important questions: what the nature of attention is (Section 2.1.1) and what collective attention is (Section 2.1.2). By answering these questions, we approach the nature of phenomena that a researcher can investigate by using the collective attention view. In the next section (Section 2.2), we present how we specifically apply the collective attention view to conceptualize constructs that explain innovation productivity in open-source software projects.

2.1.1 The Nature of Attention

We regard the attention literature in the fields of psychology, neuroscience, philosophy, management, and information systems as a reference system to define attention. Our purpose is not to offer a comprehensive review of this extensive multi-disciplinary literature but to explain important decisions we make for developing the definition for our study. These decisions enable us to contextualize the concept of attention to the innovation phenomenon in the open-source software development context. They also determine which aspects of innovation are relevant in the collective attention view. Thus, a researcher naturally foregrounds and backgrounds certain elements when applying the collective attention view to study innovation in the open-source context. Specifically, we define *attention* as a subject's selection of items for actions in a behavior space of many candidate items*.*

Decision 1: *Anchor the functional role of attention at selection*. This first decision is based on a convergent starting point of the attention literature: James' original idea on selectivity of attention (James 1890). In other words, attention functions when a subject confronts the problem of selecting a limited number of items from the enormous number of items available for the subject to process. The existence of choice space is the premise. If there is only one item for the subject to process, attention is not relevant.

From the perspective of innovation in open-source software development, this first decision regarding the definition of attention foregrounds the selective retention and backgrounds the variation generation during innovation. This is important to acknowledge as variation generation and selective retention are both considered to be fundamental to innovation (e.g., Campbell 1960; Benner and Tushman 2015; Simonton 2013). According to our first decision regarding the definition of attention, we limit the scope to selective retention. Although variation generation leads to items (e.g., preliminary thoughts to be developed or specific work to be done) that constitute the premise of attention, it is in selection retention that attention functions. Selective retention corresponds to the phenomenon that only part of the generated items are chosen to be further developed into innovation outputs.

Therefore, the collective attention view proposed by this dissertation does not fit innovation phenomena that concern broadening up variations (i.e., items to be selectively retained). Consider research that views open-source development model as a sourcing strategy (e.g., Ågerfalk and Fitzgerald 2008). The underlying value proposition of knowledge sourcing is that getting access to more ideas on variation about the extant innovation benefits innovation. However, the collective attention view fits innovation phenomena stressing that innovators have too many ideas on variations and need to cautiously make selections.

Principle 1: The collective attention view foregrounds selective retention and backgrounds variation generation during innovation in open-source software development.

Decision 2: *View attention as selection for actions*. Following Wu's (2011 and 2014) *selection for action* view of attention, we focus on the role of attention in behaviors and accordingly conceptualize attention as a subject's selection of an item for the purpose of guiding actions. By situating attention within the agency (i.e., concrete constraints from action), the selection for action view assumes that agency necessitates attention (Neumann 1987). This assumption moves beyond the consensus that attention is necessitated by cognitive limitations. Cognitive limitations lead to the conception of attention as, for example, selection that is required to avoid information overload (e.g., load theory by Lavie 2005; attention deficit trait by Hallowell 2005).

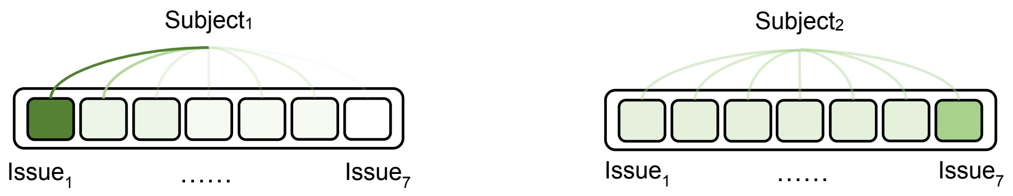
The agency assumption is not to deny the existence of cognitive limitations. It argues that "independent of all (cognitive) capacity considerations, selection is evidently needed for the control of action" (Neumann 1987, pp. 374). The central problem that forces attention on the scene is "how to avoid the behavioral chaos that would result from an attempt to simultaneously perform all possible actions" (Neumann 1987, pp. 374). For example, the reason why a developer selects only one issue to work on in a day may not be that the developer is not capable of mentally processing multiple issues during the day. The reason may be that the developer has limited computing capacity (from available equipment) to carry out the processing of multiple issues concurrently.

The second decision regarding the definition foregrounds a behavior space and backgrounds a cognition space for innovation to depict the selectivity of attention. Both spaces potentially draw a canvas that manifests the mapping path between many possible responses and many input items. The difference is that selection in the behavior space is necessitated not only by cognitive limitations but also by other contextual and environmental conditions. A selected mapping path, such as a thought of product design, can hover at the cognition layer and never land to any action at the behavior layer (e.g., creating a prototype) due to, for example, the lack of time or effector systems. Therefore, the collective attention view, which foregrounds a behavior space to depict the selectivity of attention, is meaningful to discern innovation phenomena that place salience on contextual or environmental conditions (e.g., limited work time) that necessitate selection. However, it is not effective in discerning innovation phenomena that background these constraints and focus on ideation or brainstorming (e.g., Potter and Balthazard 2004).

Principle 2: The collective attention view foregrounds a behavior space and backgrounds a cognition space to depict the selectivity of attention during innovation in open-source software development.

Decision 3: *Characterize attention by the distribution pattern of a subject's actions in a behavior space*. This third decision aligns with Mole's (2010) view of attention as an adverbial phenomenon that is best, and most fundamentally, explained by reference to the manner in which something is happening. This contrasts with a process-first phenomenon that is best and most fundamentally explained by reference to the specific type of process they instantiate. Drawn on this understanding of the metaphysical category to which attention belongs, the collective attention view proposed by this dissertation does not force the selection of actions to pre-defined inputs. Instead, it allows a subject to respond to any or all of the input items to varying extents, i.e., granular actions on many items. As such, it depicts the essence of a subject's attention (i.e., selective mental engagement with the many input items) as the distribution of the subject's granular actions in a behavior space.

For instance, subjects facing multiple open issues for software development can exhibit different attention in how they respond to the issues. In a behavioral perspective, we can contrast their attention based on the allocation of actions to issues: the more attentive a subject is to an issue, the more actions the subject allocates to the issue. Looking at Figure 2.1, assume that the first three issues are feature requests while the others are bug reports. Since subject$_1$ allocates more actions to issue$_1$ while subject$_2$ allocates more actions to issue$_7$, we know that subject$_1$'s attention is more feature oriented while subject$_2$'s attention is more bug oriented. Meanwhile, we can contrast how focused the subjects are by characterizing the dispersion of their actions over the issues. Since subject$_1$'s action distribution is more concentrated than subject$_2$'s, we know that subject$_1$'s attention is more focused than subject$_2$'s attention. It is easy to imagine many other possible distribution patterns of actions to issues. The variation in the distribution of actions to issues across subjects indicates variation in subjects' attention.



***Notes**: The behavior space is framed by one dimension: issues. The first three issues are feature requests, while the other issues are bug reports. Actions distribute along this dimension. Darker color corresponds to more actions by a subject to an issue.*

Figure 2.1 Moving Toward Attentive Patterns

The third decision regarding the definition foregrounds the granular actions for trials of innovation ideas and backgrounds the processes or mechanisms that lead to selection decisions on innovation ideas. The collective attention view thus does not fit innovation phenomena in which the innovator spends a significant amount of effort freezing the design of innovation and then, takes action based on the frozen design points. The selectivity of attention in those phenomena is strategic and punctuated and thus, is best and most fundamentally explained by specific processes or mechanisms that lead to selection decisions on innovation ideas, e.g., Apple’s creative selection over iterative demos (narrated by Kocienda 2018). However, the collective attention view fits innovation phenomena that accommodate multiple trials of innovation ideas and allow adjustments to occur during the innovation process. The selectivity of attention in these phenomena incrementally emerges as the innovative work unfolds and thus, is best and most fundamentally explained by the distribution pattern of granular actions in a behavior space. The more attentive the innovator is to an innovation idea, the more actions the innovator conducts for the innovation idea.

Principle 3: The collective attention view foregrounds the granular actions for trials of innovation and backgrounds the specific processes or mechanisms that lead to selection decisions for innovation in open-source software development.

2.1.2 Collective Attention

Extending the conceptualization of attention from the individual level to the collective level, we define *collective attention* as a collective's selection for actions in a behavior space. The collective is comprised of all the individual developers involved in a release cycle of an OSS project. Accordingly, we take a gestalt perspective to conceptualize collective attention. The maxim of the gestalt perspective is that "the whole is more than the sum of its parts" (Sternberg and Sternberg 2012, Chapter 1, page 13; Wertheimer and Riezler 1944). Mapping to attention, we should understand a collective's attention not by characterizing each individual's attention but by characterizing how individuals assemble their attention together and function as a collective. In other words, what matters more is not how each individual allocates attention but how the collective assembles the attention of individuals together as a whole.

For instance, prior literature (Bansal et al. 2018) argues that paying attention to a broad range of measurement enables organizations to notice latent issues in large-scale processes. Consider two organizations (viewed as two collectives) that pay attention to the same range of measurement but differ in the way they assemble individuals' attention to the measurement. One collective chooses to differentiate individuals' attention and concentrates each one's attention on a narrow range of measurement. In contrast, the other one chooses to overlap individuals' attention and stretches each individual's attention to a broad range of measurement.

Given individuals’ bounded rationality, it is likely that the former outperforms the latter. In other words, the way a collective assembles its individuals’ attention can make a difference.

Two underlying assumptions are thus injected into the collective attention view. First, linking to granular actions, a collective’s selection cannot be fully understood by characterizing each individual’s distribution of actions to items but need to be understood by characterizing the distribution of actions by all individuals in the collective. By taking the actions of all individuals into account, this assumption regards all individuals as stakeholders of a collective’s selection and requires the comprehensive observability of individuals’ actions. Despite the stringent requirement on observability, it relaxes the theoretical view’s reliance on the existence or effectiveness of managerial force, which underpins prior literature (e.g., Vuori and Huy 2016; Li et al. 2013; Barnett 2008; Ocasio 1997; Simon 1947) that regards (top or middle) managers as the stakeholder of an organization’s selection and observes managers’ attention allocation. Since individuals’ granular actions for open-source software development are observable, and there is no significant managerial force, this assumption facilitates the contextualization of attention view to open-source software development.

Second, attention allocation of individuals in a collective is interdependent. Consider the interdependency in collective attention partition. Let X and Y be two items (e.g., open issues) requiring attention from a collective with two individuals, A and B. If the collective tends to partition individuals’ attention over items, A’s actions will be concentrated on one item, and B’s actions will be concentrated on the other. On the contrary, if the collective tends to integrate individuals’ attention to items, actions of individuals will be concentrated on either item X or item Y, meaning that A and B will have overlapped attention to items. Note that, to surface how two collectives differ in their individuals’ attention interdependency, two dimensions, i.e., items and individuals, are used to frame the behavior space in which the distribution of a collective’s actions manifests. We can access regularities regarding a collective's attention allocation, i.e., selection for actions, by conceptualizing the distribution of a collective's actions in this multi-dimensional behavior space.

This conceptualization is especially useful for discerning phenomena on open-source software development, where regularities of collective attention allocation are hidden and emerge from individuals' interactions. While collective composition is fluid and allocation of individuals' attention (selection of what to work on) is unsupervised, prior literature on open-source software development indicates that regularities exist in the collective work. Without a planned modular software design architecture, superposition (i.e., sequential layering of individuals' work) functions as the dominant work orchestration mechanism for collaboration in open-source software development (Howison and Crowston 2014; Medappa and Srivastava 2019). As individuals' work builds on top of each other over time, their attention allocation to the required work is influenced by their own, as well as others' previous attention allocation. Thus, regularities of collective attention allocation are hidden and emergent in individuals' selection for granular actions within a behaviors space of required work.

Besides, without hierarchical authority or commands, OSS projects use arm's length mechanisms enabled by digital platforms and associated practices to coordinate work interdependencies (Lindberg et al. 2016; Ma and Agarwal 2007; Ransbotham and Kane 2011; Zammuto et al. 2007). These mechanisms take place primarily by any individuals' execution of platform-based activities such as commenting, reviewing or labeling. Cues (such as comments, code reviews, or labels) generated by the activities nudge individuals' attention allocation but do not force their allocation or significantly change their incentives as hierarchies can do. Moreover, the sources of these cues are heterogeneous. While it is difficult to theorize regularities of collective attention allocation by articulating selection processes or mechanisms underlying the cues, the essence of collective attention allocation can be effectively characterized as the distribution pattern of actions in a multi-dimensional behavior space.

Principle 4: In the collective attention view, a collective's attention allocation is effectively characterized by the distribution pattern of all its individuals' actions within a multi-dimensional behavior space for innovation in open-source software development.

2.2 APPLYING THE COLLECTIVE ATTENTION VIEW TO DEPICT THE ORGANIZATION OF OPEN-SOURCE SOFTWARE DEVELOPMENT

Applying the collective attention view, we develop four constructs (i.e., attention partition and attention augmentation by issues and by modules) to characterize collective attention allocation in a given release cycle of an OSS project. We take three steps to conceptualize these four constructs. First, we identify two modes for organizing individuals' work to achieve collective innovation productivity in prior innovation literature. They are a division mode and an integration mode. Second, we accommodate the two seemingly contradictory organizing modes by distinguishing an individual's primary and secondary focus. We conceptualize *attention partition* by depicting how a collective differentiates its individuals' primary foci and *attention augmentation* by depicting how the collective overlaps its individuals' secondary foci. Finally, by further distinguishing the orientation of collective attention allocation (issue or module), we conceptualize attention partition (or attention augmentation) by issues and by modules. Table 2.1 summarizes the definitions and operationalization of the four constructs, i.e., attention partition and attention augmentation by issues and by modules.

Table 2.1 Constructs to Describe Collective Attention in Open-Source Software Development

Construct	Definition	Operationalization
Attention partition by issues	The extent to which individuals' primary foci on issues differ in a release cycle of an OSS project	• Identify developers who have conducted innovative work (or made commits) for project i in release cycle t; • Regard the issue(s) at the highest rank of a developer's innovative work (i.e., with the most lines of code change) as the developer's primary focus; • Calculate Blau's index for the diversity of developers' primary foci.
Attention partition by modules	The extent to which individuals' primary foci on modules differ in a release cycle of an OSS project	• Identify developers who have conducted innovative work (or made commits) for project i in release cycle t; • Regard the module(s) at the highest rank of a developer's innovative work (i.e., with the most lines of code change) as the developer's primary focus; • Calculate Blau's index for the diversity of developers' primary foci.
Attention augmentation by issues	The extent to which individuals support each other's innovative work on issues with a secondary focus in a release cycle of an OSS project	• Identify developers who have conducted innovative work (or made commits) for project i in release cycle t; • Regard issues that are not at the highest rank of a developer's innovative work (i.e., with the most lines of code change) as the developer's secondary focus; • Average the proportion of each developer's secondary focus that overlaps other developers' primary or secondary foci.
Attention augmentation by modules	The extent to which individuals support each other's innovative work on modules with a secondary focus in a release cycle of an OSS project	• Identify developers who have conducted innovative work (or made commits) for project i in release cycle t; • Regard modules that are not at the highest rank of a developer's innovative work (i.e., with the most lines of code change) as the developer's secondary focus; • Average the proportion of each developer's secondary focus that overlaps other developers' primary or secondary foci.

2.2.1 Attention Partition and Attention Augmentation

The innovation literature delineates two fundamental modes for organizing individuals' work to attain collective innovation productivity, namely, a division mode and an integration mode. Although constructs developed in the organization or formal team context for characterizing the two organizing modes cannot be directly generalized to the context of open-source software development, rationales underlying these two modes are insightful for conceptualizing collective attention in the context of open-source software development.

For a **division mode**, innovative work is decomposed into loosely coupled clusters and delegated to individuals (e.g., Raveendran et al. 2016; Ethiraj and Levinthal 2004; Sanchez and Mahoney 1996). This mode accelerates innovation production by facilitating deep exploitation and broad exploration for avenues of innovation (Yayavaram and Ahuja 2008; Baldwin and Clark 2000). Highly useful innovations often emerge from deep exploitation and broad exploration (Katila and Ahuja 2002). Besides, given humans' bounded rationality and limits on cognitive processing, allocating clusters of innovative work to different individuals rations the attention of the collective (Simon 1997). Individual can focus on managing the interdependencies among elements within the cluster allocated to them, at the expense of interdependencies with elements in other clusters due to loose couplings.

Mapping to collective attention, the division mode suggests that a collective differentiates its individuals' attention over tasks in the required innovative work. Tasks can be characterized with respect to issues or modules, and they are items requiring the collective's attention. In other words, for a collective that adopts a division mode in a release cycle of an OSS project, individuals' innovative actions in the release cycle should exhibit minimal overlap on tasks.

For an **integration mode**, the objective is to synthesize the ideas or work of different individuals (e.g., Lingo and O'Mahony 2010; Harvey 2014). The integration mode helps to build connections among people by either introducing disconnected individuals or facilitating new coordination between connected individuals. The connections facilitate coordination, collaboration, and pursuit of common goals in collective innovation (Obsteld 2005), which accelerates the collective's production of innovation outputs. Besides, different individuals can have divergent perspectives on problems and specific actions to solve them (Cronin and Weingart 2007; Heracleous and Barrett, 2001). The synthesis of divergent perspectives often involves identifying and challenging existing assumptions and the prevailing paradigm, which enables the generation of breakthrough ideas and exemplars of these ideas (Harvey 2014). As a result, novel outputs are likely to be created.

Mapping to collective attention, the integration mode suggests that a collective overlaps its individuals' attention over tasks in the required innovative work. In other words, for a collective that adopts an integration mode in a release cycle of an OSS project, individuals' innovative actions in the release cycle should exhibit high overlap on tasks.

To reinvestigate rationales underlying the two organizing modes in the open-source software development context, we apply the collective attention view to conceptualize constructs that depict the extent to which a collective organizes its individuals' work in a division or an integration mode (i.e., the extent to which individuals' attention differs or overlaps with respect to their allocation of actions to tasks). Besides, different from prior literature that regards the division and integration modes as two ends of a spectrum (e.g., more or less decomposability by Yayavaram and Ahuja 2008), we regard them as two distinct constructs that can co-exist and can be combined in different ways. Specifically, we accommodate the two seemingly contradictory modes by distinguishing between tasks that are within the scope of an individual's attention as follows: (i) *primary focus,* which refers to the task on which an individual's innovative actions concentrate, and (ii) *secondary focus,* which refers to all other tasks within an individuals' scope of attention.

Building on the above distinction, we conceptualize **attention partition** as the extent to which individuals' primary foci differ in a release cycle of an OSS project. It discerns the extent which individuals in a collection concentrate or divide their primary attention. For a collective with high attention partition, we will observe individuals' primary foci are dispersed over tasks for the required innovative work; in contrast, for a collective with low attention partition, individuals' primary foci are concentrated on tasks. We align the division mode with individuals' primary foci for two reasons. First, a considerable number of individuals allocate their attention to more than one task, given that their attention allocation is unsupervised. Moreover, to ration attention, they allocate a high proportion of attention to their primary foci. Second, prior literature (Howison and Crowston 2014) has found that the majority of work in an OSS project is accomplished with one individual working on one task. Thus, we infer that dividing, instead of integrating, individuals' primary foci can be important for innovation productivity.

Next, we conceptualize **attention augmentation** as the extent to which individuals support each other's innovative work with a secondary focus in a release cycle of an OSS project. It speaks to the integration mode achieved through individuals' secondary foci. For a collective with attention augmentation, we will observe that individuals' secondary foci overlap with others' attention (i.e., with others' primary or secondary foci). Aligning the integration mode with individuals' secondary foci makes it possible to consider that division and integration modes can co-exist and can be combined in different ways. A collective where individuals' primary foci are differentiated and where individuals' secondary foci are overlapped with the attention of others has both attention partition and attention augmentation, exhibiting a dual focus. In contrast, a collective can also exhibit only attention partition or attention augmentation, or neither.

2.2.3 Issue and Module Orientation

After conceptualizing attention partition and attention augmentation, we further distinguish a collective's attention allocation orientation. Tasks within the required innovative work call a collective's attention, but how to conceptualize the decomposition of required innovative work into tasks? Two reference systems readily exist in the collective's publicly shared innovation agenda: the list of issues and the modularity of the codebase. The former regards innovative work required for solving the same issue as a task (or a cluster of tasks), and it places salience at activities for problem-solving. In contrast, the latter regards innovative work required for modifying or creating the same module as a task (or a cluster of tasks), and it places salience on objects to be generated. Prior literature (Raveendran et al. 2016) indicates that placing salience at different aspects of the required work can impact innovation output production.

For attention partition, a collective can implement it with a one-way or two-way orientation. Take attention partition by issues as an example of the one-way orientation. If a collective has attention partition by issues but not by modules, its individuals' primary foci on issues will differ, but their primary foci on modules will overlap. The individuals will exhibit diversity in the issues on which they concentrate their innovative actions but will exhibit overlaps in the modules on which they concentrate their innovative actions. However, if the collective has used both reference systems for attention partition, its individuals' primary foci on both issues and modules will differ. The individuals will exhibit diversity in both the issues and modules on which they concentrate their innovative actions.

Similarly, for attention augmentation, a collective can also implement it with a one-way or two-way orientation. If a collective has attention augmentation by issues but not by modules, its individuals' secondary foci on issues will largely overlap with others' attention to issues, but their secondary foci on modules will not overlap with others' attention to modules. A large proportion of issues to which individuals allocate relatively fewer innovative actions (compared with issues on which they concentrate their innovative actions) will also be the issues to which others allocate innovative actions, but only a small proportion of modules to which individuals allocate relatively fewer innovative actions will also be the modules to which others allocate innovative actions. However, if the collective has used both reference systems for attention augmentation, its individuals' secondary foci on both issues and modules will largely overlap with others' attention to issues and modules.

As one-way and two-way orientation to tasks can manifest for attention partition or attention augmentation, we elaborate attention partition and attention augmentation based on orientation to issues or modules. This enables us to evaluate how the resulting four constructs can be combined to manifest as different configurations in collective attention allocation.

2.3 A CONFIGURATIONAL PERSPECTIVE OF COLLECTIVE ATTENTION FOR INNOVATION PRODUCTIVITY

Since our theoretical focus is uncovering how the four collective-attention constructs can combine to affect innovation productivity outcomes, we adopt a configurational perspective. Furthermore, we consider contingencies that can intertwine with collective attention allocation. The inclusion of the contingencies enables us to assess whether effective configurations of collective attention vary with contingencies. The contingencies are related to the scope of innovative work, the availability of development resources, and the temporal aspects of release cycle (i.e., project maturity and length of release cycle).

2.3.1 Collective-Attention Configurations and Innovation Productivity

We adopt a configurational perspective to understand the relationships between collective-attention constructs and innovation productivity. Our choice is guided by past work in IS and management that has sought to understand how the patterns and combinations of elements are related to outcomes (e.g., El Sawy et al. 2010; Fiss 2011; Park et al. 2020). Specifically, our choice is based on three reasons, as we now explain.

First, the configurational perspective, compared with the variance perspective, better approaches the pattern (or gestalt) nature of collective attention in open-source software development. Constructs under the umbrella of collective attention characterize the collective-level order (or pattern) of the same distribution from distinct aspects. As the distribution is emergent from individual-level innovative actions and the allocation of those actions is unsupervised, researchers need to understand the actions simultaneously to derive constructs that characterize the pattern. If the actions cannot be understood separately, the constructs should better be understood as a whole.

Second, the four collective-attention constructs are likely to have complex causal relationships with the outcome in forms of equifinality, multifaceted causality, as well as causal asymmetry. In terms of equifinality (or the presence of multiple ways to success), we expect that distinct combinations of collective attention constructs can yield similar outcomes, e.g., “high-speed”, “high-novelty”, or “both high-speed and high-novelty” collective innovation. In terms of multifaceted causality, we expect that the four constructs can suppress, substitute, or complement each other’s effect. In terms of causal asymmetry, we expect that combinations representing “high-speed” collective innovation are not simply a mirror image of combinations representing “low-speed” collective innovation but differ fundamentally. Neither are “high-novelty” and “low-novelty” combinations.

Third, the effect of collective attention combinations is intertwined with contingent conditions related to the scope of innovative work, the availability of development resources, and the temporal aspects of release cycle. Those conditions are highly subject to chance, since developers on the platform are free to work on any OSS project on the platform at any time. As a consequence, those collectives are constrained in planning their allocation of collective attention. Their allocation is more likely to be ad hoc and emerges from their adaptive adjustments to relevant contingencies. The adaptive and emergent nature determines that the effect of collective attention combinations is inseparable from those contingencies. Thus, we need to also study the collective-attention constructs and contingencies as combinations.

2.3.2 Contingent Configurations and Intertwined Contingencies

We expect that the sufficiency of collective-attention configurations varies with contingent conditions related to the scope of innovative work, availability of development resources, and temporal aspects of release cycle. Note that the configurational approach does not expect researchers to include all elements that are connected to the outcome. It expects researchers to propose elements that play an important role in determining whether the outcome in question presents or not. For instance, among the numerous elements related to firm performance, Fiss (2011) includes eight elements (related to organization structure, strategy, and environment) to investigate configurations for high and very-high firm performance. Campbell et al. (2016) include nine elements to investigate configurations that elicit a “good deal” and “bad deal” as perceived by market participants.

The complexity of configurational analysis grows exponentially as researchers add potential elements into the discovery of configurations. A favorable configurational framework includes a reasonable number of elements as a holistic combination. Furthermore, combinations of those elements (in terms of whether each element presents or not) have reasonable coverage of cases that show the specific outcome, e.g., the both-high-speed-and-high-novelty outcome. Thus, we balance parsimony and coverage and aim not to comprehensively include all contingent conditions that are connected to the outcome of interest but to specify some that are likely to be most relevant in the collective attention view.

First, we consider contingent conditions related to scope of innovative work. The scope of innovative work can be depicted by the number of issues and the number of modules involved in a focal release cycle of an OSS project. A broad scope of innovative work is relevant to innovation productivity because it increases the probability of generating a useful or a breakthrough innovation output. Besides, a broad scope of innovative work indicates that the collective's selection space on attention partition and attention augmentation (by issues or by modules) is large. In contrast to a limited scope of innovative work with few issues or few modules, the effective configurations of collective attention may be different.

Second, we consider contingent conditions related to the availability of development resources. We measure the availability by the number of developers involved in a focal release cycle of an OSS project. Similar to a broad scope of innovative work, having many developers working on the software development is likely to be beneficial for the speed of producing software updates and the novelty of produced software updates. To reduce coordination cost and to build connections among individuals, attention partition and attention augmentation can be more or less important when the collective has many developers than when it has few.

Finally, we consider contingent conditions related to key temporal aspects of a release cycle. Prior open-source software literature (e.g., Ren et al. 2016; Dahlander and O'Mahony 2011) finds that OSS projects are self-organizing systems that can evolve toward their high-performing zones. Thus, collective attention allocation intertwines with the timeline to impact the innovation productivity of the collective during the focal release cycle. Accordingly, we focus on project maturity, measured by the number of days elapsed before the start of the focal release cycle. In addition, we consider the length of the release cycle as it can also affect the efficacy of the collective attention constructs.

To conclude, Figure 2.2 presents our theoretical framework with both collective attention constructs and contingencies under consideration.

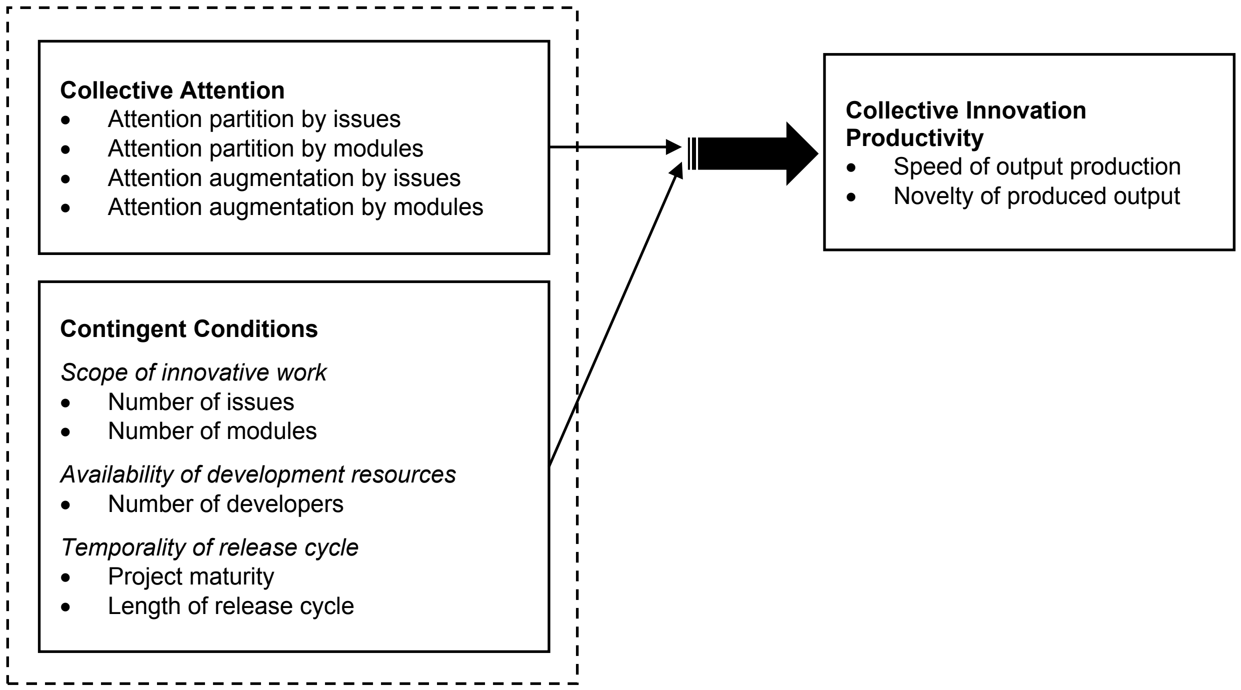

Collective Attention
• Attention partition by issues
• Attention partition by modules
• Attention augmentation by issues
• Attention augmentation by modules
Contingent Conditions
Scope of innovative work
• Number of issues
• Number of modules
Availability of development resources
• Number of developers
Temporality of release cycle
• Project maturity
• Length of release cycle
Collective Innovation Productivity
• Speed of output production
• Novelty of produced output

Figure 2.2 Theoretical Framework on Configurations of Collective Attention and Intertwined Contingencies for Innovation Productivity

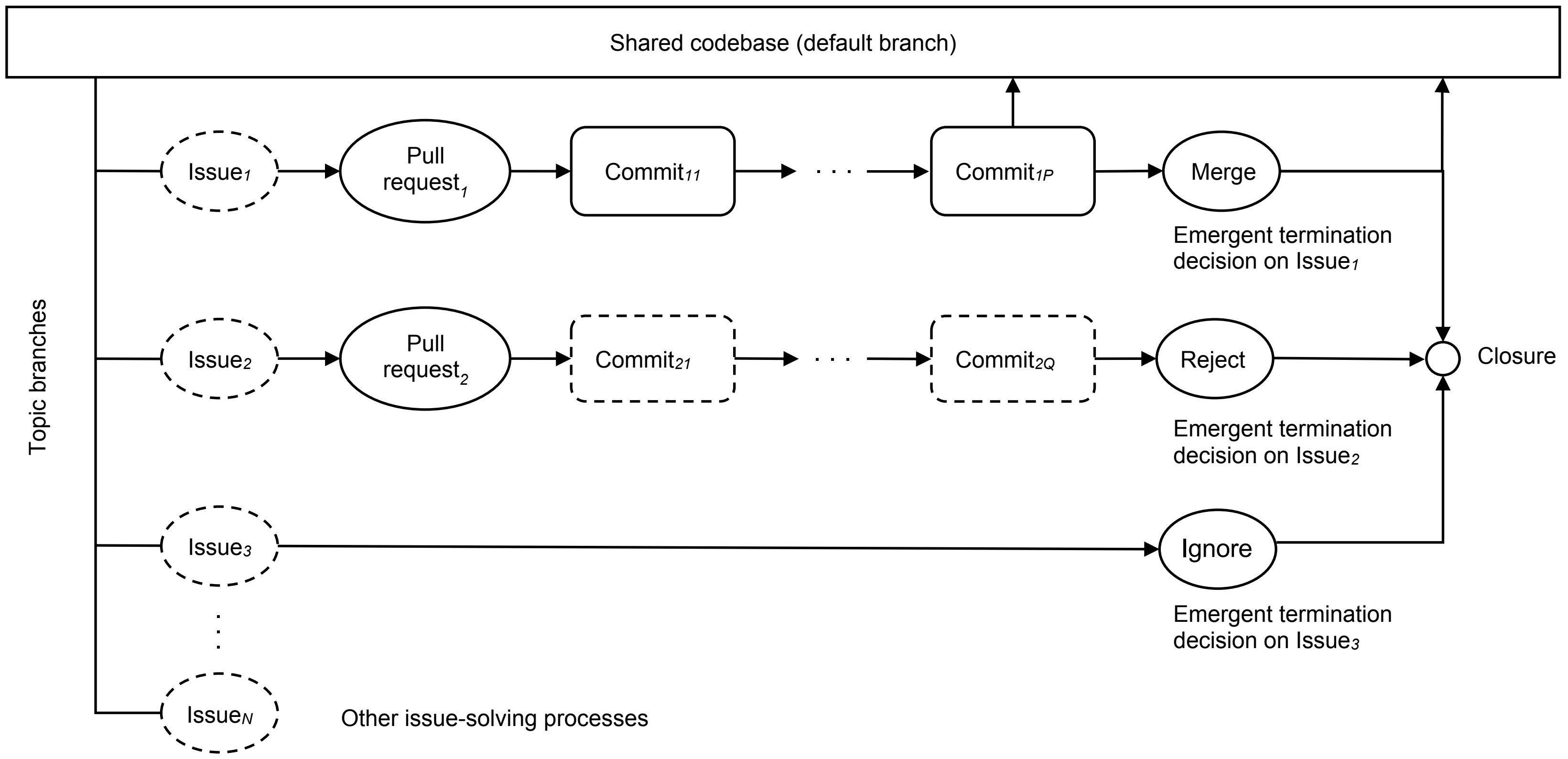
3. METHODS

We conducted our empirical investigation in the GitHub context, an online platform where developers from all over the world build software together. The observability of comprehensive granular innovative actions (or commits) and the rich functionalities that enables self-organization make GitHub an appropriate empirical setting for us to study the relationship between collective attention allocation and innovation productivity.

While software projects on GitHub can be private for a team, the public ones, i.e., open-source software (OSS) projects, are our focus. For these projects, any developer can propose issues (e.g., bugs or feature requests) within the software and make changes in the codebase to resolve these issues by opening pull requests (or issue-solving processes), as shown in Figure 3.1. Commits occur when developers upload code changes along pull requests. Each commit refers to an innovative action, which is our unit of observation. If a commit, or a series of commits, leads to a valid solution that meets the project's vision, the solution is merged into the shared codebase as an intermediate innovation output (or a software update such as a patch or a new feature). Our unit of analysis is a release cycle of an OSS project.

Developers who work in the same release cycle constitutes a collective for it. GitHub offers rich functionalities for the collective to self-organize the allocation of collective attention. For instance, the collective can use the assigning functionality to assign and de-assign issues to individuals, which enables the partitioning of collective attention. The collective can also use the mentioning functionality to draw an individual's attention to an issue, even if the issue is not the individual's primary focus, thereby enabling the augmentation of collective attention. Meanwhile, the collective can use issue-connecting and labeling functionality to restructure proposed issues based on modules that need to be changed to resolve issues, thereby enabling module-oriented allocation of collective attention.

As collectives use the platform-based functionalities and associated practices to allocate their collective attention differently, we collect data and construct a sample of projects and their release cycles to empirically study how those that achieve high speed, high novelty, or both organize their attention allocation differently from the others. In the following sections, we discuss how we collect data and construct a sample for the empirical investigation (Section 3.1), what analytical approach we use to do the configurational analyses (Section 3.2), and how we measure and calibrate the variables of interest (Section 3.3).


Shared codebase (default branch)
Topic branches
Issue$_1$
Pull request$_1$
Commit$_{11}$
Commit$_{1P}$
Merge
Emergent termination decision on Issue$_1$
Issue$_2$
Pull request$_2$
Commit$_{21}$
Commit$_{2Q}$
Reject
Closure
Emergent termination decision on Issue$_2$
Issue$_3$
Ignore
Emergent termination decision on Issue$_3$
Issue$_N$
Other issue-solving processes

Notes**: Open-source software development in GitHub projects (or repositories) includes continuously identifying and solving issues. Developers from the community open pull requests and make commits (or innovative actions) to introduce changes that resolve issues within the shared codebase. Commits linked by arrows represent a series of activities that together form the process of solving an issue. There are three types of innovative actions:* ***in-progress *(white boxes in solid line),* ***merged*** *(grey boxes in solid line) and* ***dropped*** *(white boxes in dashed line). An issue-solving process can be terminated in three ways: merging the solution into the shared codebase, rejecting the finished or unfinished solution, or ignoring the issue. An issue closed with no commits is an ignored issue.*

Figure 3.1 Snapshot of Open-Source Software Development in GitHub Projects

3.1 SAMPLE AND DATA

We collected data in several steps. First, we used GitHub APIs (application programming interfaces) to fetch a list of all “machine-learning” OSS projects (or public root[1] repositories in GitHub’s terminology). We scoped our empirical study to one specific topic (i.e., machine learning) to control for the influences of factors (e.g., science) at levels higher than the collective level. We chose the “machine-learning” topic for two reasons: (i) machine learning is a high-tech space, where the number of OSS projects grow exponentially on GitHub (as shown in Figure 3.2), making the pursuit of “high speed”, “high novelty”, or “both high speed and high novelty” relevant goals for projects on the topic and (ii) it is one of the most popular topics on GitHub, allowing us flexibility to impose additional sample-selection criteria without losing sample diversity. We obtained a list of 36,280 machine-learning project repositories created from January 2008 (when GitHub was launched) to December 2019. Our data collection for a repository is from its creation until August 2020. As such, we had at least eight months’ data for each repository in the sample.

For 2,539 repositories with a shared codebase and software releases, we fetched the basic information of the repositories as well as their associated 328,443 pull requests and 2,681,018 commits. For each pull request, we identified all associated commits by tracing commit-uploading events along this thread of conversation. For each commit, we used GitHub APIs to fetch information on the developer who made (or authored) the code changes and information on files that had been changed through additions or deletions. We identified developers by their user login and identified modules by the folder (or the path to) of changed files. Files changed in

[1] Root repositories refer to repositories that are not copied (or forked in GitHub’s terminology) from other repositories.

a commit could be within one module or across modules. The identifiable association among commits, developers, pull requests, and modules makes it feasible for us to operationalize collective attention constructs, which depict the distribution pattern of innovative actions in the behavior space framed by developers, issues, and software modules.

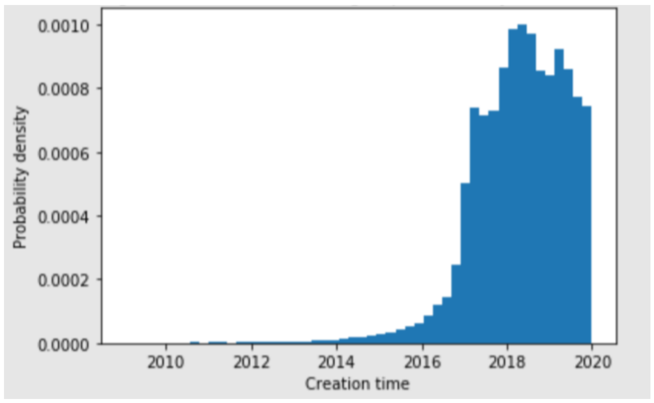


Figure 3.2 Histogram of Machine-Learning Open-Source Software Projects Created on GitHub Over the Years

Consequently, we excluded repositories irrelevant to our research inquiry. Our research concerns the pursuit of speed and novelty of collective innovation in OSS projects. Therefore, we excluded repositories that had limited software development activities on GitHub since when they were created. Specifically, we removed 2,082 repositories with less than 40 (near 75 percentile of the 2,539 repositories) pull requests and less than 20 (near 75 percentile of the 2,539 repositories) developers who had ever made a commit in the repository's history. We also removed 77 repositories under any one of the three conditions: (i) having less than one year history, (ii) having a very small set of source code files (less than 180,000 bytes, which was near 50 percentile of the 2,539 repositories), or (iii) having the first release published even before the repository was created on GitHub. After imposing these repository-level selection criteria, we obtained a sample of 380 repositories.

At the level of release cycles, we excluded those that do not align with our research inquiry. First, the emergence of collective-level pattern needs time. If a release cycle is too short, we cannot tell whether the distribution of innovative actions captures the hidden regularity of a collective or is just a random case. Thus, we removed release cycles that were too short, i.e., less than 14 days (50 percentile of the 380 repositories' 7,238 release cycles). Although prior empirical study by Howison and Crowston (2014) indicates that the release cycles of OSS projects are approximately 45 days in their sample, we find release cycles on GitHub tend to be much shorter and set the threshold as 14 days. Second, given our research interest in a collective's innovation productivity, we removed release cycles with only one developer involved in the software development. Third, we removed release cycles with missing information due to data-fetching limitations of GitHub. After our screening of the release cycles based on these considerations, we obtained a sample of 3,052 release cycles from 363 GitHub machine-learning projects.

3.2 ANALYTICAL APPROACH

We use a set-theoretic approach, fuzzy-set qualitative comparative analysis (fsQCA), to do the analyses. This analytical technique is uniquely suited for our research. It is based on a configurational understanding of how causes combine to bring about a specific outcome of interest, and it can handle significant levels of causal complexity (Fiss, 2007; Ragin, 2000, 2008). Besides, as causes and outcomes that interest us vary by degree or levels, we adopt fuzzy-set theory by using fsQCA instead of crisp QCA, which requires all variables to be binary.

Intuitively speaking, this approach views each data point as a case, and it views combinations (or configurations) of causes and the outcome as fuzzy sets of cases with membership levels (ranging from 0 to 1). Then, it infers the causal sufficiency and necessity relationships between the configurations and the outcome by investigating the fuzzy-subset relationship between them. If the empirical evidence supports that a configuration is a fuzzy subset of the outcome, the configuration is sufficient to elicit the outcome. In other words, it is a solution configuration for the outcome. After identifying all such configurations, this technique simplifies the solutions and evaluates their overall and unique explanation power. To conclude, our analytical approach includes four processes: (i) building fuzzy sets for investigation, (ii) discerning solution configurations, (iii) simplifying solution expression, and (iv) evaluating explanation power of solution configurations.

3.2.1 Building Fuzzy Sets for Investigation

The empirical investigation of fsQCA focuses on the relationship between configurations of causes and a specific outcome of interest. To understand the relationship, we need to build fuzzy sets of all possible configurations of causes and a fuzzy set of the outcome. Each configuration is a combination of causes that indicate whether their characteristics are present or absent. Thus, all possible configurations of k causes can be expressed as a truth table with 2^k rows. For example, our four collective-attention constructs constitute 16 possible configurations of collective attention allocation, depicting whether a collective partitions and augments its attention by issues and by modules. These possible configurations are candidate solution configurations for the outcome.

To investigate which ones are solution configurations, we need to first fit our data to the possible configurations and the outcome. As the causes and outcomes that interest us vary by degree or level, our data do not fit neatly to the configurations and the outcome. We adopt fuzzy-set theory (Zadeh 1965) and build fuzzy sets of cases with membership scores ranging from 0.0 (non-membership) to 1.0 (full membership). The core of this step is set calibration, which starts with clear specification of a target set. Assume that our target set is a configuration of attention partitioned by issues but not by modules and attention augmentation both by issues and by modules, expressed as $pai * {\sim}pam * aui * aum$, where $\sim$ means absence and $*$ means logical AND. Before obtaining each case's membership in this configuration, we need to calculate its membership in the sets of four conditions in the configuration.

We adopt a well-established calibration method proposed by Ragin (2000, 2008) to transform scale values of each condition into fuzzy membership scores. This method structures calibration with three important anchors: the threshold for full membership, the threshold for full non-membership, and the cross-over point, i.e., the point of maximum ambiguity (i.e., fuzziness) in the assessment of whether a case is more in or out of a set. Using the three anchors, this method translates the original data into the metric of log odds and converts log odds to membership scores by applying the standard formula: $degree\ of\ membership\ = exp(log\ odds)/[1 + exp(log\ odds)]$. We accomplish these computational steps by using a simple *compute* command in the software package fsQCA 3.0 (Ragin and Davey 2016).

After obtaining cases' membership score for each one of the four conditions (pai, pam, aui, and aum), we use fuzzy-set operations (i.e., logical AND, OR, and Negation) to calculate their membership score to the configuration of these conditions. We first use Negation to calculate each case's membership score to ${\sim}pam$, which equals 1 minus each case's membership in

pam. Using logical AND, we then calculate each case's membership in the configuration, which takes the minimum membership score of each case's membership scores to pai, $\sim pam$, aui, and aum. For example, if a collective's membership in the set of pai is 0.60, its membership in the set of pam is 0.11, its membership in the set of aui is 0.75, and its membership in the set of aum is 0.80, then its membership in $pai * \sim pam * aui * aum$ is 0.60.

Similarly, we can obtain each case's membership score to all other possible configurations and its membership score to the outcome. Excluding very few cases with membership score to all possible configurations as 0.5, we can assign each case into one possible configuration with membership score above 0.5. Recall that each row of the truth table refers to a possible configuration. The empirical cases can be sorted into rows of the truth table, with some rows containing many cases, some rows just a few, and some rows containing no cases. Since the credibility of analytical results for a configuration decreases as the number of cases falling into the configuration decreases, we need to impose a *frequency threshold*, i.e., the minimum number of cases required for a configuration to be considered as a candidate solution for the outcome. Prior studies (e.g., Fiss 2011; Campbell et al. 2016; Park and Mithas 2020) have set the frequency threshold as low as three.

3.2.2 Discerning Solution Configurations

After specifying configurations under investigation and building fuzzy-sets for these configurations and the outcome, the next step is to investigate which configurations are solution configurations, namely, sufficient to elicit the outcome. The investigation relies on the fuzzy-subset relation between the fuzzy set of a configuration and the fuzzy set of the outcome. As shown in the left plot of Figure 3.3, configuration $\boldsymbol{X}$ is a subset of the outcome $\boldsymbol{Y}$; all $\boldsymbol{X}_i$ values are less than or equal to their corresponding $\boldsymbol{Y}_i$, where i refers to a specific case in the data, $\boldsymbol{X}_i$ refers the case's membership in configuration $\boldsymbol{X}$, and $\boldsymbol{Y}_i$ refers the case's membership in the outcome $\boldsymbol{Y}$. To assess the fuzzy subset relation, we use the concept of *consistency*, defined as the proportion of cases on or above the main diagonal of the plot (Ragin 2000). The consistency score is 1.0 (100 percent consistent) if all the cases plot on or above the main diagonal of the plot (as shown in the left plot of Figure 3.3).

Turning to measures, one straightforward measure of set-theoretic consistency using fuzzy membership scores is the sum of the *consistent* membership scores in a configuration divided by the sum of *all* the membership scores in the configuration (Ragin 2003; Ragin 2008: page 51). This measure can be further refined, and we use a refined measure introduced by Ragin (2006), which gives small penalties for minor inconsistencies and large penalties for major inconsistencies. The adjustment is accomplished by adding to the numerator the part of each inconsistent membership score that is consistent with the outcome. This consistency measure is computed as:

$$Consistency(\boldsymbol{X}_i \leq \boldsymbol{Y}_i) = \frac{\sum[\min(\boldsymbol{X}_i, \boldsymbol{Y}_i)]}{\sum(\boldsymbol{X}_i)}$$

where min means the selection of the lower one of the two values. For discerning solution configurations, the convention is that if the consistency is greater than 0.8, an investigator claims that configuration $\boldsymbol{X}$ is "almost always" sufficient (or a solution) to elicit the outcome $\boldsymbol{Y}$, and the acceptable benchmark is 0.75 (Ragin 2006, 2008).

In addition to sufficiency, the fuzzy subset relation helps us to understand the necessity of a specific condition in the configuration. As shown in the right plot of Figure 3.3, if it is the opposite that the outcome $\boldsymbol{Y}$ is a subset of condition $\boldsymbol{X}$, meaning all $\boldsymbol{Y}_i$ values are less than or equal to their corresponding $\boldsymbol{X}_i$, condition $\boldsymbol{X}$ is necessary for eliciting the outcome $\boldsymbol{Y}$. As the inequality signaling necessity ($\boldsymbol{Y}_i \leq \boldsymbol{X}_i$) is the reverse of the inequality signaling sufficiency ($\boldsymbol{X}_i \leq \boldsymbol{Y}_i$), the consistency measure of the subset relationship for a necessary condition is:

$$Consistency(\boldsymbol{Y}_i \leq \boldsymbol{X}_i) = \frac{\sum[\min(\boldsymbol{X}_i, \boldsymbol{Y}_i)]}{\sum(\boldsymbol{Y}_i)}$$

A consistency score of 1.0 means that each case's membership in $\boldsymbol{Y}$ is less than or equal to the corresponding membership in $\boldsymbol{X}$.

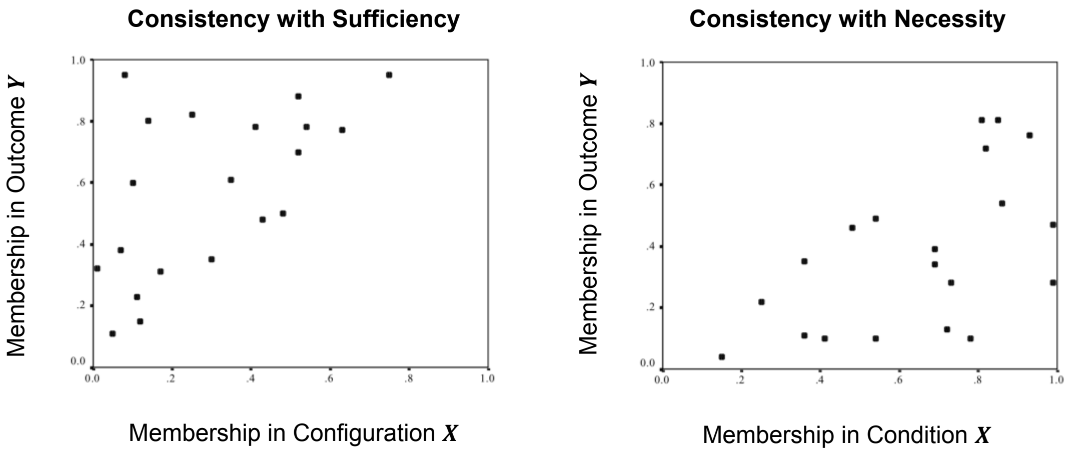


Figure 3.3 Fuzzy Subset Relation Consistency

3.2.3 Simplifying Solution Expression

After identifying solution configurations, we can reduce the complexity of configurational solutions by using an algorithm based on Boolean algebra. The purpose is to simplify the interpretation of configurational results. We use the truth table algorithm described by Ragin (2005, 2008), and the algorithm is embedded in the fsQCA software. It produces three solutions:

complex, parsimonious, and intermediate solution. The *complex* solution is obtained by using a fundamental rule in Boolean algebra. The rule is that if two configurations differ in only one causal condition yet produce the same outcome, then this causal condition is redundant and can be removed to create a simpler expression. For instance, if the combination $A * B * C$ (read: A, B, and C) and the combination $A * {\sim}B * C$ (read: A and C but not B) both lead to the outcome of interest, then the combination $A * C$ produces the outcome.

The *parsimonious* solution is obtained by doing counterfactual analysis of causal conditions, which enables us to categorize causal conditions into core and peripheral causes. Counterfactual configurations have no or very few cases falling into them, and they are removed by imposing the frequency threshold when building fuzzy sets for investigation. This situation happens due to limited diversity of naturally occurring phenomena. We do not have empirical evidence to assess whether these counterfactual configurations would lead to the outcome of interest or not. However, in the counterfactual analysis, the algorithm assumes that they elicit the outcome. If a condition in configurations of the complex solution cannot be reduced even if we include these assumptions, it is a *core* element; otherwise, it is a *peripheral* element.

The *intermediate* solution is obtained by distinguishing “easy” and “difficult” counterfactuals (see Ragin 2008 or Fiss 2011 for details) and including only “easy” counterfactuals to further simplify the complex solution. While both easy and difficult counterfactuals do not have empirical evidence to support whether they lead to the outcome or not, but the uncertainty of easy counterfactuals can be addressed by assumptions based on theoretical or substantial knowledge. These assumptions speak to the link between each condition and the outcome. As an example, assume one has evidence that the combination $A * B * {\sim}C$ leads to the outcome. No evidence exists as to whether the combination $A * B * C$ would also lead to the outcome, but theoretical and substantial knowledge links the presence of C to the outcome. In such a situation, the combination of $A * B * C$ is an easy counterfactual. After incorporating the easy counterfactual, the expression can be simplified as $A * B$.

Different from parsimonious solution, which includes all counterfactuals to simplify the complex solution, the intermediate solution uses only the easy counterfactuals. It sits between the complex solution and the parsimonious solution and demonstrates a hybrid of empirical evidence and theoretical or substantial knowledge. Understanding the benefits of imposing assumptions to distinguish easy and difficult counterfactuals, we choose not to do so. The reason is that prior literature has not offered strong evidence to support whether the presence or the absence of collective-attention constructs that interest us benefits or harms the goals of achieving high speed, high novelty, or both high speed and high novelty. Thus, in our analytical results, the intermediate solution is the same as the complex solution. Elements occur both in the parsimonious solution and complex solution are core causes, while those occur only in the complex solution are peripheral causes.

3.2.4 Evaluating Explanation Power of Solution Configurations

The final step is to evaluate the explanation power of configurations in the complex (or intermediate) solution. The evaluation relies on investigating the relevance of a solution configuration against the outcome. To understand the relevance, we use the concept of set-theoretic *coverage*, which indicates the empirical importance of a causal configuration (Ragin 2008). As the analytical approach we use allow for equifinality (Mackie 1965; George 1979; George and Bennett 2005) and causal complexity (Ragin 1987), a general finding is that a given outcome may result from several different configurations of conditions. Therefore, in addition to the importance of each solution configuration (coverage of a configuration), we want to understand the importance of these solution configurations as a whole (overall coverage) and the relative importance of each solution configuration (unique coverage of a configuration).

The Venn Diagram in Figure 3.4 illustrates the coverage concepts in fuzzy-sets, with the areas in the figure representing proportions against the outcome. Configuration $\boldsymbol{X}$ and configuration $\boldsymbol{Z}$ are both subsets of outcome $\boldsymbol{Y}$, i.e., sufficient to elicit the outcome $\boldsymbol{Y}$. Areas I, II, and III together indicate the overall coverage of $\boldsymbol{X}$ and $\boldsymbol{Z}$. The two solution configurations, as a whole, are relevant, because the unexplained area IV is small. Specific to the coverage of each configuration, area I and area II together indicate the part of $\boldsymbol{Y}$ explained by configuration $\boldsymbol{X}$, and area II and area III together indicate the part of $\boldsymbol{Y}$ explained by configuration $\boldsymbol{Z}$. Area I indicates the unique coverage of $\boldsymbol{X}$, and area III indicates the unique coverage of $\boldsymbol{Z}$. As the unique coverage of $\boldsymbol{X}$ is noticeably greater than that of $\boldsymbol{Z}$**,** we know that configuration $\boldsymbol{X}$ is more important than configuration $\boldsymbol{Z}$ on explaining the outcome $\boldsymbol{Y}$.

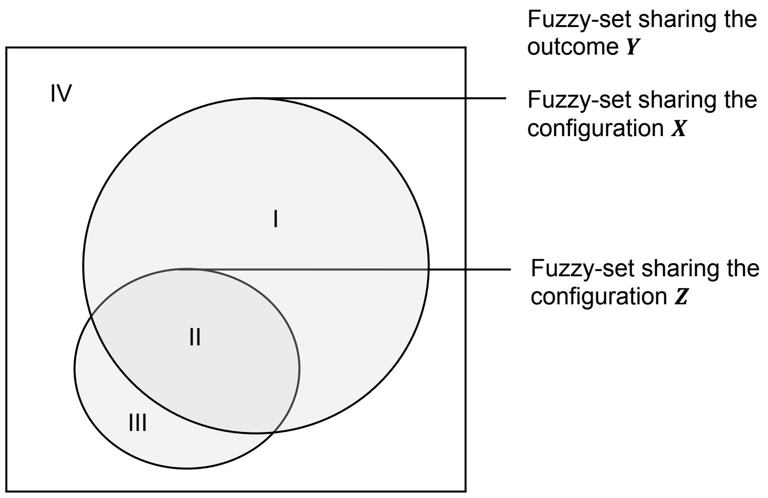

Fuzzy-set sharing the outcome $\boldsymbol{Y}$
IV
Fuzzy-set sharing the configuration $\boldsymbol{X}$
I
Fuzzy-set sharing the configuration $\boldsymbol{Z}$
II
III

Figure 3.4 Venn Diagram Illustrating the Concept of Coverage in Fuzzy Sets

As to the fuzzy-set measures of coverage, the coverage of a configuration, e.g., $\boldsymbol{X}$, is simply the overlap expressed as a proportion of the sum of the membership scores in the outcome (Ragin 2008: Chapter 3):

$$Coverage(\boldsymbol{X}_i \leq \boldsymbol{Y}_i) = \frac{\sum[\min(\boldsymbol{X}_i, \boldsymbol{Y}_i)]}{\sum(\boldsymbol{Y}_i)}$$

In the same logic, the overall coverage of multiple solution configurations, e.g., $\boldsymbol{X}$ and $\boldsymbol{Z}$, is:

$$Coverage(\boldsymbol{X}_i \leq \boldsymbol{Y}_i\ or\ \boldsymbol{Z}_i \leq \boldsymbol{Y}_i) = \frac{\sum\{min[max(\boldsymbol{X}_i, \boldsymbol{Z}_i), \boldsymbol{Y}_i]\}}{\sum(\boldsymbol{Y}_i)}$$

where max takes the greater one of two values. One can infer from the above two formulars that the overall coverage of two configurations should be greater or equal to the coverage of each configuration. In the same vein, the formular for the unique coverage of a configuration, e.g., $\boldsymbol{X}$, is:

$$Unique\ coverage\ of\ \boldsymbol{X} = \frac{\sum\{min[max(\boldsymbol{X}_i, \boldsymbol{Z}_i), \boldsymbol{Y}_i]\}}{\sum(\boldsymbol{Y}_i)} - \frac{\sum[\min(\boldsymbol{Z}_i, \boldsymbol{Y}_i)]}{\sum(\boldsymbol{Y}_i)}$$

which indicates the difference between the overall coverage of two solution configurations ($\boldsymbol{X}$ and $\boldsymbol{Z}$) and the coverage of the other solution configuration ($\boldsymbol{Z}$).

3.3 MEASURES AND CALIBRATION

Before zooming into specific measures, let us explain how we identify a release cycle (our unit of analysis) and a collective (who owns the problem of collective attention allocation) in open-source software development. Since the development is continuous, we use the period between the timestamps of two releases to identify a release cycle. In other words, the start time of a focal release cycle is when the previous release is published (if it is an initial release, the start time is when the repository is created), and its end time is when the current release is published. As to a collective, it refers to developers who are involved into the software development during a focal release cycle. Individuals of the collective use platform functionalities and the associated practices to coordinate the allocation of their attention. The structure of the collective is flat, and there is no significant managerial force supervising the allocation of individuals' attention.

3.3.1 Measuring Innovation Productivity

Innovation productivity gauges the amount of innovation outputs produced per unit of input. Specific to our research, the innovation outputs are intermediate innovation outputs for software development, i.e., software updates (such as new features or patches for bugs) merged into the shared codebase. As we are interested in the speed of producing innovation outputs, a unit of input can be understood a unit of time, e.g., a day. We measure the *speed of output production* by the number of software updates merged per day during a release cycle of an OSS project. Note that not all pull requests that a collective opens to solve issues lead to an intermediate innovation output. Consider the situation that the collective rejects to merge useless updates or terminates a pull request in the middle of the issue-solving process when sensing that the code changes in progress will not lead to a useful update.

As to the *novelty of the produced output*, we measure it by the average lines of code change introduced by the intermediate innovation outputs. Novelty refers to the quality of being new. We choose the software itself, instead of other software on the market, as the reference system to assess the newness of an intermediate innovation output. As to the degree of newness, lines of code change (additions plus deletions) can work as a reasonable proxy. We use it to gauge the extent to which an output makes the updated software deviates from its previous version. A collective can have high speed but low novelty by generating many minor software updates or have low speed but high novelty by generating few major software updates. The speed and novelty measures speak to different aspects of innovation productivity.

3.3.2 Measuring Collective Attention Constructs

The operationalization of four collective attention constructs includes three steps. The first two steps are common, but the third step of measuring attention partition (by issues and by modules) and that of measuring attention augmentation (by issues and by modules) differ. First, recall that collective attention constructs characterize the distribution pattern of innovative actions. The first step is to identify innovative actions conducted in a release cycle (a unit of analysis) of an OSS project. These innovative actions refer to commits made along pull requests of the OSS project within the period of the focal release cycle. Based on the association between commits and developers, we identify a list of involved developers. They constitute a collective involved in the release cycle.

Second, note that attention partition depicts how individuals’ primary foci differ whereas attention augmentation depicts how their secondary foci overlap with others’ attention. The second step is to identify a developer’s primary and secondary focus. A closer investigation on how a developer’s innovative work is distributed over issues and over modules enables us to do so. We regard an issue (or a module) at the highest rank of the developer’s innovative work as this developer's primary focus. Specifically, the ranking is based on lines of code change that the developer has contributed to each issue (or module) by making commits during the release cycle. Since the size of (or amount of changes included in) commits varies dramatically, we do not gauge the amount of innovative work by the number of commits. If multiple issues (or modules) are at the highest rank, they together indicate the developers' primary focus. All other issues (or modules) the developer has worked on during the release cycle are at the developers' secondary focus. We rank the amount of innovative work by observing how the lines of change are distributed over issues (or modules).

For attention partition, the third step is to calculate the Blau's index (Blau 1977; Harrison and Klein 2007) regarding the diversity of individuals' primary foci in terms of issues and modules. Take attention partition by issues as an example. Attention partition by modules just follows the same calculation rules. Regarding issue as a categorical variable, we first categorize developers based on which issues are their primary foci. A developer will be assigned to more than one issue if multiple issues are this developer's primary focus. In these cases, we "split" the developer equally to these multiple issues. As an example, assume that two issues are at one's primary focus. Then, each issue has 0.5 developer falling into it. We calculate the proportion of developers whose primary focus is on issue i and denotes the proportion as P_i. Attention partition by issues can be measured by $1 - \sum_{i=1}^{M} P_i^2$, where M is the number of issues involved in individuals' primary foci.

For attention augmentation, the third step is to calculate the extent to which individuals' secondary foci overlap others' scope of attention. Similarly, let us take attention augmentation by issues as an example. Attention augmentation by modules just follows the same calculation rules. An individual's scope of attention regarding issues covers all issues on which this individual has worked during a release cycle. After identifying issues that are included in an individual's secondary focus, we calculate the proportion of issues that overlap others' scope of attention (i.e., issues that are included not only in the focal individual's secondary focus but also in others' primary or secondary foci) and denotes the proportion as P_i. Then, attention augmentation by issues can be calculated as the average proportion of all individuals, i.e., $\frac{1}{D}\sum_{i=1}^{D} P_i$, where D is the total number of developers in the collective.

3.3.3 Measuring Contingent Conditions

We operationalize the *scope of innovative work* by two measures: 1) the number of issues (or pull requests) on which a collective has worked during a focal release cycle of an OSS project and 2) the number of modules on which the collective has worked during the release cycle. We measure *availability of development resources* by the number of developers who have conducted innovative actions during the release cycle. As to the temporal aspect, we measure *project maturity* by the number of days that have elapsed before the release cycle starts and after the OSS project is created. We measure the *length of the release cycle* by the number of days the release cycle has, i.e., days between the timestamp of the last release and the current release.

3.3.4 Calibration

The variables in our study vary by degree or levels. We need to transform them into set measures. As mentioned in section 3.2.1, the calibration method that we use needs the specification of three anchors (full membership, full non-membership, and cross-over point) for each variable. We used the 75th percentile, the 25th percentile, and the median of the sample to set the three anchors for each variable. We understand the limitations of using this sample-dependent calibration strategy and the advantages of using well-established external anchors, which defines the qualitative breakpoints of full membership, full non-membership, and cross-over points of a variable. However, we do not have such well-established anchors. The collective-attention constructs and their measures are created by us, and for the contingency and outcome variables that interest us, there are no consistent external standards.

What we have is a diverse and relatively large sample (3,052 release cycles that reasonably span the range of each variable), which enables us to approach the conventions of open-source software development pertaining to our variables of interest. Besides, we addressed the limitation of sample-dependent calibration by diligently conducting sensitivity analysis. Consistent with Fiss (2011), we varied the cross-over point between +/-25 percent for all variables. Specifically, we started with varying the collective-attention variables and then the contingency variables. Solution configurations identified in the main analyses are well supported in the sensitivity analyses. Minor changes regarding the pattern of solution configurations are observed, but the interpretation of the results remains substantively unchanged. For elaboration of the results, see the robustness analyses section of Chapter 4 (Subsection 4.1.2 and Subsection 4.2.2, Robustness 2: Sensitivity analyses).

4. ANALYSES AND RESULTS

The focus of our analyses is to discern the sufficiency of candidate configurations for three innovation productivity goals—i.e., speed, novelty, or both. As shown in Table 4.1, we conducted three sets of analyses (main, robustness, and exploratory) to discover and consolidate solution configurations for achieving a singular goal of high speed or high novelty and for achieving the dual goal of both high speed and high novelty.

The purpose of the main analyses is to discern solution configurations for achieving the three innovation productivity goals. Based on the research design in Chapter 3 to collect data, construct the sample, measure and calibrate variables, we analyzed the relationships between the causal conditions and the outcomes of interest. Table 4.2 and Table 4.3 show the descriptive statistics and correlation matrix of all variables. Our sample covers a reasonable range for each of the causal and outcome variables; the four collective-attention constructs are empirically distinguishable (with correlations < 0.60).

Next, to address potential issues with respect to the robustness of the findings, we conducted the following **robustness analyses**.

First, we used an alternative approach that scholars suggested for studying configurations (or profiles) of causes, i.e., the use of deviation score (Doty et al. 1993; Drazin and Van de Ven 1985). Here, we regard a solution configuration as an ideal type and cases' Euclidean distances to the configuration as deviation scores between the "ideal" and empirically observed profiles. We use the deviation scores to examine how a case's fit to (or deviation from) the ideal types affects the production of (or membership to) a specific outcome. Greater deviation from a solution configuration should result in lower membership to the outcome, whereas better fit to the ideal types should result in higher membership to the outcome.

Second, we tested the sensitivity of our findings in the main analyses to our calibration strategy. Consistent with Fiss (2011), we varied the cross-over point between +/- 25 percent for all variables. With the newly calibrated variables, we check the change of solution configurations and whether the consistency scores of solution configurations in the main analyses decrease to unacceptable levels. For example, we take raw consistency into consideration when identifying solution configurations in the main analysis. The raw consistency of a solution configuration should be no less than 0.8, i.e., the convention of being "almost always" sufficient to elicit the outcome, and the acceptable benchmark is 0.75 (Ragin 2006, 2008). If the raw consistency of a solution configuration drops to a level much less than 0.75 after varying the cross-over point between +/-25 percent, the solution configuration was sensitive to calibration. In other words, the configuration was not a robust solution for the outcome.

Third, we measured two module-oriented collective-attention constructs (i.e., attention partition by modules and attention augmentation by modules) at the file level and investigated whether solution configurations in the main results remain substantively unchanged. In the main analyses, we operationalized these two constructs at the folder level. This operationalization assumes that the modularity happens at the folder level. However, collectives for open-source software development are also very likely to implement modularity at other levels, e.g., the file level. Prior literature has indicated that the modularity level can impact innovation performance in such complex systems (Ethiraj and Levinthal 2004). Thus, we conducted file-level operationalization of two module-oriented collective-attention constructs to check the robustness of solution configurations in the main results.

Fourth, we adjusted our definition of an individual’s primary focus, a core concept for the conceptualization of collective attention constructs. In the main analyses, we assume that only issues or modules at the highest rank (with most lines of code change) are at an individual’s primary focus. All other lower-ranked items are considered to be the individual’s secondary focus. This assumption has face validity, as in our empirical study we observe that an individual’s work on issues or modules at the highest rank dominates the individual’s work in a focal release cycle (on average, 82.95% for issues and 67.17% for modules). To investigate the change of solution configurations when the assumption is relaxed, we also include items at the second highest rank of innovative work as part of an individual’s primary focus. With new measures on collective-attention variables using this revised definition of an individual’s primary focus, we re-investigate configuration solutions for the three innovation productivity goals.

Fifth, we scoped the types of issues only to those pertaining to feature requests. This robustness analysis is motivated by some practitioners’ opinion that open-source software development is organized around feature requests (Haddad and Warner 2011). These practitioners tend to background innovative work regarding bug fixing and documentation updating, although prior literature (e.g., Howison and Crowston 2014) has demonstrated that these two types of work, together with the work on feature requests, constitute three general types of task in open-source software development. Specifically, we used the text-classification technique to classify issue-solving processes (or pull requests) into three categories: feature requests, bug fixing, documentation updating. Then, we scoped to innovative work (or commits) on feature requests to measure our variables of interest and investigate whether the resultant solution configurations are consistent with those in the main analyses.

Sixth, we excluded individuals at the periphery of a focal collective, i.e., developers who made less than 15 lines of code change during a focal release cycle of an OSS project. This final robustness analysis is motivated by the observation that a considerable proportion of individuals are at the periphery and contribute very few lines of code changes, e.g., less than 15 lines. What if we adjust our definition of a collective for open-source software development in a focal release cycle by excluding those peripheral individuals? To seek the answer, we scope our analyses to innovative work conducted by individuals who have made at least 15 lines of code change (25th percentile of code contribution made by developers) during the focal release cycle. Similar to previous robustness analyses, we then re-investigate solution configurations for three productivity goals by using the new measures and checking whether the solution configurations remain substantively unchanged.

Finally, we conducted **exploratory analyses** pertaining to two additional contingencies regarding the composition of individuals in a focal collective:

- We consider collective diversity with respect to *tenure diversity* as a contingency. Prior literature (e.g., Ren, Chen and Riedl 2016) has found that group diversity affects group productivity in online open collaboration by using data on Wikipedia projects.
- We consider the presence of *star contributors* in a collective. Prior literature (e.g., Aguinis and O'Boyle 2014) has suggested that having star contributors is beneficial for collective work characterized by increased complexity, reduced situational constraints (such as geographic distances and inability to access information) and flexible hierarchies. Open-source software development has such features.

As such, we evaluated whether the inclusion of these contingencies improves our results in terms of empirical relevance (higher coverage being more desirable), parsimony (fewer, parsimonious configurations being more optimal than a large number of configurations), and sense-making (more fundamental insights on contingent solutions of collective attention being preferred).

Table 4.1 Summary of Analyses Conducted

Analyses	Purpose	Sample	Technique
Main analyses	Discern solution configurations for achieving high speed, high novelty, and both high speed and high novelty	**Sample 0**: 3,052 release cycles of 363 repositories	fsQCA[*]
Robustness 1: Deviation score analyses	Test whether the effect of solution-configuration deviation (fit) on membership to the corresponding outcome is significantly negative (positive)	**Sample 0**: 3,052 release cycles of 363 repositories	Two-limit Tobit regression model[◊]
Robustness 2: Sensitivity analyses	Test the sensitivity of solution configurations to calibration by varying the cross-over point +/-25 percent for all variables	**Sample 0**: 3,052 release cycles of 363 repositories	fsQCA[*]
Robustness 3: File-level analyses	Test whether the solution configurations hold if we operationalize attention partition and attention augmentation by modules at file (instead of folder) level	**Sample 0**: 3,052 release cycles of 363 repositories	fsQCA[*]
Robustness 4: Redefining primary focus	Test whether the interpretation of main results remains substantially unchanged if we redefine primary focus as issues (or modules) at not only the highest but also the second highest rank of innovative work	**Sample 1**: 3,052 release cycles of 363 repositories	fsQCA[*]
Robustness 5: Feature-only analyses	Test whether the interpretation of main results remains substantively unchanged if we scope to innovative work on feature requests	**Sample 2**: 2,134 release cycles of 350 repositories	fsQCA[*]
Robustness 6: Scoping involved developers	Test whether the interpretation of main results remains substantially unchanged if we exclude individuals at the periphery of a collective, i.e., ones who have made less than 15 lines of code change	**Sample 3**: 2,813 release cycles of 362 repositories	fsQCA[*]
Explorative analyses: Including additional contingencies	Explore the possibility that the addition of other contingencies may improve the results in terms of empirical relevance (higher coverage being more desirable), parsimony (fewer, parsimonious configurations being more optimal than a large number of configurations), and sense-making	**Sample 0**: 3,052 release cycles of 363 repositories	fsQCA[*]

Notes: [*] The term fsQCA is short for fuzzy-set qualitative comparative analysis, and we use fsQCA 3.0 software (Ragin and Davey 2016). [◊] We use the `tobit` command of Stata 14. Note that except for the sensitivity analyses, we used the same sample-dependent calibration strategy across all robustness analyses, i.e., using 75th, 50th, and 25th percentiles to set the full membership, cross-over, and full non-membership anchors. Besides, across all robustness analyses, we imposed the same sample selection criteria that we imposed in the main analyses; see Appendix A for descriptive statistics and correlation matrices for variables in Samples 1-3.

In the following three sections, we elaborate our analytical results. We take **two steps** to decipher the relationship between collective-attention configurations and the three innovation-productivity goals. In the **first step**, we do not take contingencies into consideration. Note that the complexity increases exponentially when we include more and more conditions into configurational analyses. The inclusion of contingencies will not generate too much additional value if we can find solution configurations for each productivity goals by considering only the four collective-attention constructs and if those solution configurations have reasonably high explanation power and pass all robustness tests. Otherwise, the **second step**, in which we take contingencies into consideration and decipher contingent solutions of collective attention for innovation productivity, will be critical. We present the results of the two steps in Section 4.1 and 4.2, respectively. In the final section (Section 4.3), we present the results of explorative analyses with the inclusion of the two additional contingencies pertaining to the composition of the collectives (i.e., tenure diversity and star contributors).

Table 4.2 Descriptive Statistics for Variables

	Collective Attention Conditions				**Contingent Conditions**					**Productivity**	
	Attention partition by issues	Attention partition by modules	Attention augmentation by issues	Attention augmentation by modules	Number of issues	Number of modules	Number of developers	Project maturity	Length of release cycle	Speed	Novelty
Obs.	3052	3052	3052	3052	3052	3052	3052	3052	3052	3052	3052
Mean	0.763	0.680	0.125	0.538	61.111	70.409	15.243	674.943	96.802	0.769	5332.98
Std.	0.190	0.215	0.164	0.276	155.386	102.549	31.776	607.077	180.195	1.315	78506.53
Min	0.000	0.000	0.000	0.000	1.000	1.000	2.000	0.000	14.000	0.000	0.000
25th	0.667	0.517	0.000	0.371	8.000	15.000	3.000	231.500	25.000	0.107	127.164
50th	0.800	0.722	0.058	0.588	21.000	34.000	6.000	530.000	45.000	0.286	405.091
75th	0.909	0.840	0.202	0.750	55.250	81.250	13.000	935.000	98.000	0.763	1261.810
Max	0.996	0.984	1.000	1.000	3032.000	1285.000	439.000	3780.000	3444.000	14.377	4124507

Notes: Descriptive statistics are based on uncalibrated measures. The number of observations for each variable is 3052 (see Obs.).

Table 4.3 Correlation Matrix of Variables

Variable	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1. Attention partition by issues	1.000										
2. Attention partition by modules	0.596	1.000									
3. Attention augmentation by issues	0.111	0.117	1.000								
4. Attention augmentation by modules	0.271	0.211	0.412	1.000							
5. Number of issues	0.314	0.287	0.105	0.170	1.000						
6. Number of modules	0.360	0.405	0.186	0.237	0.510	1.000					
7. Number of developers	0.367	0.362	0.119	0.121	0.768	0.580	1.000				
8. Project maturity	0.129	0.129	-0.006	0.009	0.155	0.164	0.238	1.000			
9. Length of release cycle	0.161	0.082	0.005	0.009	0.143	0.145	0.067	-0.133	1.000		
10. Speed	0.368	0.359	0.150	0.261	0.474	0.500	0.564	0.218	-0.141	1.000	
11. Novelty	-0.005	0.032	0.018	0.005	-0.009	0.066	0.005	-0.008	-0.004	-0.007	1.000

Notes: Correlations are based on uncalibrated measures. We use the standard correlation coefficient (the Pearson correlation coefficient).

4.1 CONFIGURATIONS OF COLLECTIVE ATTENTION FOR INNOVATION PRODUCTIVITY

In this section, we consider only collective-attention constructs when doing analyses. After presenting the main results, we check the robustness of solution configurations in the main results across the six robustness analyses.

4.1.1 Main Results

Four collective attention constructs together produce 16 possible configurations in terms of whether the characteristic of a construct is present or absent. Table 4.4 shows how the over 3,000 empirical cases (release cycles) distribute over these possible configurations with membership scores above 0.50. The configuration with the smallest number of cases is a combination with the absence of issue-oriented but the presence of module-oriented attention partition and attention augmentation. It has 40 cases, which is much greater than the frequency thresholds used in prior literature (such as eight by Campbell et al. 2016 and three by Fiss 2011). Thus, we do not impose a frequency threshold to exclude any configuration that can be a potential solution. It means that we do not have counterfactual configurations. Besides, twelve configurations have over 100 cases. Overall, we find the four collective-attention constructs are compatible as causes in the same theoretical framework, and they are effective for characterizing collectives for open-source software development.

As to the outcomes, we considered three productivity goals, i.e., high speed, high novelty, both high speed and high novelty. Moreover, to investigate whether there is causal asymmetry between collective attention and innovation productivity, we also considered the negation of three productivity goals. They are unfavorable outcomes, i.e., low speed, low novelty, low speed or low novelty). For sufficiency and necessity analyses, we need to establish consistency thresholds on the relationship between collective attention and the outcomes. For necessity analyses, a consistency benchmark of at least 0.90 is recommended, and a high coverage measure is required to indicate that the potential necessary condition is empirically relevant (Ragin 2008; Schneider and Wagemann 2012; for an empirical example, see Greckhamer 2016). Although the coverage levels of all four collective-attention constructs (in terms of presence) are high (no less than 0.63), none (in terms of presence or absence) is necessary. The highest consistency level is 0.73 (for the presence of attention partition by issues and by modules), less than 0.90.

For consistency analyses, a well-established consistency benchmark is at least 0.80 for raw consistency, and the acceptable level is 0.75 (Ragin 2000, 2008). We followed the convention and set 0.80 as the threshold. Except for raw consistency, it is important to consider PRI (proportional reduction in inconsistency) scores, a more exacting measure of consistency, to avoid simultaneous subset relations of configurations in both the outcome and its negation (e.g., both high speed and low speed). While there is no well-established benchmark for PRI consistency, the general rule is that it should be high and ideally not too far from raw consistency scores, e.g., 0.70, and that configurations with PRI scores below 0.50 indicate significant inconsistency. We set 0.70 as the threshold and observe the gaps between configurations' PRI scores. We adjusted the threshold slightly down (0.69 or above) if a salient gap occurred after a PRI score approximating 0.70 (e.g., 0.696).

Based on these raw-consistency and PRI-consistency thresholds, we identified solution configurations for the specific innovation productivity outcomes. The right six columns of Table

4.4 speak to the six specific innovation productivity outcomes, under which the value 1 indicates that the configuration in the same row is a solution configuration. We find solution configurations for high speed and for high novelty. Although we do not find solution configurations for the dual goal (i.e., both high speed and high novelty), we know that 13 (of 16) configurations will not lead to the dual goal because they consistently lead to low speed or low novelty. It motivates us to further consider contingencies (see Section 4.2). We expect that the remaining 3 configurations may elicit the dual goal under appropriate contingent conditions. Besides, for speed, 11 configurations fall into the ambiguous zone, where we do not have empirical evidence to determine whether a configuration benefits or hurts innovation speed. Similarly, for novelty, 13 configurations fall into the ambiguous zone. These observations further motivate us to drill into contingent solutions of collective attention allocation.

Table 4.4 Truth Table of Collective Attention and Innovation Productivity Outcomes

Configuration				**Number of cases**	**High speed**[*]	**Low speed**[*]	**High novelty**[*]	**Low novelty**[*]	**Both high**[*]	**Either low**[*]
pai	pam	aui	aum							
1	1	1	1	573	**1**	0	**1**	0	0	0
1	1	1	0	229	0	0	0	0	0	0
1	1	0	1	170	**1**	0	0	0	0	0
1	1	0	0	279	0	0	0	0	0	**1**
1	0	1	1	156	0	0	0	0	0	**1**
1	0	1	0	59	0	0	0	0	0	**1**
1	0	0	1	73	0	0	0	0	0	**1**
1	0	0	0	115	0	0	0	**1**	0	**1**
0	1	1	1	107	0	0	0	0	0	**1**
0	1	1	0	56	0	0	0	0	0	**1**
0	1	0	1	40	0	0	0	0	0	**1**
0	1	0	0	101	0	**1**	0	0	0	**1**
0	0	1	1	223	0	0	0	0	0	**1**
0	0	1	0	123	0	**1**	0	0	0	**1**
0	0	0	1	185	0	0	0	0	0	**1**
0	0	0	0	563	0	**1**	0	**1**	0	**1**

Notes: For the configuration column, pai refers to attention partition by issues, pam refers to attention partition by modules, aui refers to attention augmentation by issues, and aum refers to attention augmentation by modules. Each row under the overarching configuration column refer to a configuration with 1 indicating presence and 0 indicating absence of a condition. Values in the next column indicates the number of cases falling into a given configuration with membership scores greater than 0.50. Other six columns at the right side are productivity outcomes, under which the value 1 indicates that the corresponding configuration is a solution configuration.

After obtaining a preliminary understanding of how possible configurations connect to the different productivity outcomes, we take a closer look at the elements of the solution configurations. Table 4.5 includes simplified solution configurations for all productivity outcomes. We follow the notation applied by Fiss (2011) and subsequent research (e.g., Campbell et al. 2016), where "●" represents the presence of a condition, "⊗ " represents its absence, and a blank space indicates a "don't care" situation, meaning that a given condition can be either present or absent (i.e., it is not causally related to the outcome). The results demonstrate equifinality and causal asymmetry. For equifinality, we find multiple solution configurations for the low-speed and low-speed-or-low-novelty outcomes. For asymmetric causality, configurations for unfavorable outcomes are not simple mirror image of configurations (if any) for the corresponding favorable outcomes.

Table 4.5 Configurations of Collective Attention for Innovation Productivity

	High speed	**High novelty**	**Low speed**		**Low novelty**	**Low speed or low novelty**		
Configuration	HS1	HN1	LS1	LS2	LN1	LL1	LL2	LL3
Attention partition by issues	●	●	⊗	⊗		⊗		
Attention partition by modules	●	●	⊗		⊗		⊗	
Attention augmentation by issues		●		⊗	⊗			⊗
Attention augmentation by modules	●	●	⊗	⊗	⊗			⊗
Raw coverage	0.459	0.347	0.407	0.397	0.347	0.611	0.604	0.463
Unique coverage	0.459	0.347	0.063	0.053	0.347	0.063	0.067	0.102
Consistency	0.829	0.803	0.838	0.833	0.762	0.880	0.892	0.903
Overall solution coverage	0.459	0.347	0.460		0.347	0.815		
Overall solution consistency	0.829	0.803	0.824		0.762	0.852		
Frequency cutoff	40*	40*	40*		40*	40*		
Consistency cutoff	0.834	0.803	0.829		0.761	0.834		

Notes: * We did not impose a frequency cutoff to remove any one of the 16 possible configurations of four collective-attention conditions; 40 refers to the smallest number of cases that a given configuration has. ● refers to the presence of a condition, and ⊗ refers to the absence. As we have no counterfactual configurations and do not do counterfactual analysis, there is no distinction between core and peripheral elements.

Specific to the singular goal of high speed, a combination of attention partition by both issues and modules and attention augmentation by modules is sufficient for achieving high speed (see HS1: $pai * pam * aum$ → high speed, in Table 4.5). It means that a dual focus of collective attention, i.e., differentiating individuals' primary foci but overlapping individuals' secondary foci (or attention partition and attention augmentation) elicits high speed. Moreover, the dual focus should be module oriented (i.e., organized by modules). Turning to issue orientation, the collective just needs to partition individuals' primary foci. Specific to high novelty, a combination of attention partition and attention augmentation by both issues and modules is sufficient for achieving high novelty (see HN1: $pai * pam * aui * aum$ → high speed, in Table 4.5). It means that collective attention should have a dual focus of collective attention, with differentiated primary focus and overlapped secondary focus, regarding both issue and module orientation.

As to achieving the dual goal of both high speed and high novelty, we do not find a solution for it. However, from the three solution configurations for low speed or low novelty (see LL1, LL2, and LL3 in Table 4.5), we find that the absence of attention partition by issues, the absence of attention partition by modules, or the absence of attention augmentation by both issues and modules, will yield a low-speed-or-low-novelty outcome. It helps us to rule out 13 of 16 candidate collective-attention solutions for achieving the dual goal. For the remaining three candidate solutions, the dual focus of collective attention (i.e., differentiating individuals' primary foci and overlapping their secondary foci) should be present in issue or module orientation or in both orientations. If the dual focus presents in just one orientation, the collective needs to differentiate individuals' primary foci in the other orientation. To conclude, our analyses demonstrate that the four collective attention constructs yield one solution for each singular goal (of high speed or high novelty) but do not yield any solution for the dual goal (of both high speed and high novelty).

4.1.2 Robustness Tests

The six robustness analyses demonstrate that the collective-attention solutions in the main results (see Table 4.5) are robust. We next elaborate the results of the robustness tests.

Robustness 1: Deviation Score Analyses

We conducted deviation score analyses following the approach in prior literature (Doty et al. 1993; Fiss 2011). Regard each solution configuration as an ideal type (or profile) of collective attention allocation, we calculated each empirical case's *deviation* from (or distance to) an ideal type by using the Euclidean distance formula[2]. Besides, we used the fuzzy set measures (rather than raw measures) to create the ideal type. Accordingly, low deviation scores correspond to full membership, high deviation scores correspond to full non-membership, and medium deviation scores tie to the crossover point. If multiple ideal types (solution configurations) were discovered for a specific outcome, we calculated the *ideal types fit* by the minimum deviation across all the ideal types, according to the following formula:

$$Fit_i = -\begin{pmatrix} T \\ minD_{it} \\ t = 1 \end{pmatrix}.$$

Here, D_{it} is the distance between case i and ideal type t, and T denotes the number of ideal types. Then, we used two-limit Tobit regression to model the relationship between idea types fit (or deviation from a solution configuration) and the membership score to the outcome; in the

[2] We also estimated the deviation score to a solution configuration by using the logic AND calculation, i.e., taking the minimum of a case's memberships to all elements in the configuration, instead of the Euclidean distance. The results are substantively identical.

model specification, we allowed for intragroup correlation within the same open-source software project (or repository).

Table 4.6 illustrates the results. Model 1 shows results of deviation from the only solution configuration for high speed (i.e., HS1), thus testing whether a collective better achieves high speed if its structural features of collective attention resemble the profile identified by HS1. The coefficient is negative and significant, which indicates that closer distance to the solution configuration is associated with higher speed. Similarly, the coefficients in all other models concerning solution-configuration deviation (i.e., Model 2, Models 4-5, Model 6, and Models 8-10) are negative and significant, which indicates that closer distance to the solution configuration is associated with higher likelihood of producing the corresponding outcome. Besides, pseudo R-squares of those models evidence the relevance of those solution configurations to the outcome (with highest value as 0.31 and the lowest value as 0.11).

Model 3 shows results of overall ideal types fit across two solution configurations for low speed. The fit coefficient is positive and significant, which indicates that a collective is more likely to have low speed if its collective-attention characteristics resemble any of the two ideal types identified by solution configurations of low speed (LS1 and LS2). In the same vein, the positive and significant coefficient in Model 7 indicates that a collective is more likely to have either low speed or low novelty if its collective-attention features resemble any of the three solution configurations for the low-speed-or-low-novelty outcome. Pseudo R-squares of both models are reasonably high (with values of 0.27 and 0.41), meaning that the solution configurations overall well explained the outcomes. To conclude, results in the deviation score analyses validate solution configurations in the main results.

Table 4.6 Tobit Models of Solution-Configuration Deviation on Collective Innovation Productivity (Robustness 1)

Independent variable	Membership in high speed	Membership in high novelty	Membership in low speed			Membership in low novelty	Membership in low speed or low novelty			
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10
HS1 deviation	-0.477*** (0.030)									
HN1 deviation		-0.137*** (0.025)								
Ideal types fit (for LS1-2)			0.468*** (0.033)							
LS1 deviation				-0.455*** (0.032)						
LS2 deviation					-0.417*** (0.032)					
LN1 deviation						-0.323*** (0.027)				
Ideal type fit (for LL1-3)							0.378*** (0.030)			
LL1 deviation								-0.385*** (0.038)		
LL2 deviation									-0.412*** (0.040)	
LL3 deviation										-0.281*** (0.028)
Constant	0.955*** (0.040)	0.851*** (0.036)	0.931*** (0.027)	0.962*** (0.029)	0.921*** (0.029)	0.824*** (0.026)	1.047*** (0.018)	0.900*** (0.014)	0.919*** (0.013)	0.914*** (0.015)
F score	255.83***	154.43***	199.71***	196.65***	165.19***	147.28***	157.85***	104.16***	107.49***	103.60***
Pseudo R^2	0.3086	0.1203	0.2695	0.2778	0.2117	0.1108	0.4060	0.2661	0.2896	0.2085
Observations	3,052	3,052	3,052	3,052	3,052	3,052	3,052	3,052	3,052	3,052

Note: We used the two-limit Tobit regression model and allowed for intragroup correlation within the same open-source software project (or repository). Robust standard errors are reported in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Robustness 2-6: Robustness Tests Regarding Key Decisions in the Main Analysis

We conducted a series of robustness tests to check the sensitivity of the main results to important decisions made in the research design. Like the main analyses, these analyses use the set-theoretic approach. We check whether solution configurations in the main results are still sufficient to elicit the outcome given those tweaks (i.e., having their consistency scores equal or above threshold for sufficiency analyses). The higher a configuration's consistency scores (raw consistency score and PRI score), the stronger the argument for its sufficiency. Moreover, to get a better understanding on the influence of those tweaks, we turn to truth tables and trace original solution configurations that constitute the simplified configurations in main results. For example, the original solution configurations for high speed are: (i) a combination of attention partition and attention augmentation by both issues and modules, i.e., $pai * pam * aui * aum$, and (ii) $pai * pam * {\sim}aui * aum$ (with ${\sim}aui$ indicates the absence of attention augmentation by issues). They produce the simplified configuration HS1, i.e., $pai * pam * aum$.

The results of robustness tests are elaborated as follows. To aid elaboration, we use the following as thresholds to assess the extent to which a solution configuration is supported in a robustness test:

- *Strongly supported*: consistency scores are above the established thresholds (i.e., raw consistency score $\geq$ 0.80, and PRI score $\geq$ 0.70);
- *Moderately supported*: raw consistency score is not below the acceptable level, and its PRI score is slightly below the established threshold (i.e., 0.80 $>$ raw consistency score $\geq$ 0.75, and 0.70 $>$ PRI score $\geq$ 0.65);

- *Weakly supported*: raw consistency score is still at the acceptable level, and its PRI score is not too much below the established threshold (i.e., 0.80 > raw consistency score ≥ 0.75, and 0.65 > PRI score ≥ 0.60);
- *Not supported*: raw consistency score is below the acceptable level, or its PRI score is noticeably below the established threshold (i.e., 0.75 > raw consistency score, or 0.60 > PRI score).

Table 4.7 Robustness Analyses of Collective-Attention Configurations (Robustness 2-3)

Configuration (not simplified)				**Main result**	**Sensitivity analyses (Robustness 2)**						**File-level analyses (Robustness 3)**		
					Cross-over 25% up			**Cross-over 25% down**					
					# of	Consistencies		# of	Consistencies		# of	Consistencies	
pai	pam	aui	aum	Label	cases	Raw	PRI	cases	Raw	PRI	cases	Raw	PRI
High speed													
1	1	1	1	HS1	370	0.861	0.816	774	0.821	0.779	629	0.831	0.786
1	1	0	1	HS1	191	0.853	0.796	216	0.804	0.730	205	0.823	0.752
High novelty													
1	1	1	1	HN1	370	0.823	0.747	774	**0.776**	0.707	629	**0.777**	**0.698**
Low speed													
0	0	0	0	LS1-2	764	0.838	0.798	498	0.866	0.825	659	0.835	0.794
0	1	0	0	LS2	79	0.816	0.707	80	0.840	0.745	34	0.821	0.688
0	0	1	0	LS1	174	0.829	0.711	85	0.834	0.682	157	0.822	0.698
Low novelty													
0	0	0	0	LN1	764	**0.752**	0.700	498	**0.771**	0.710	659	**0.747**	**0.692**
1	0	0	0	LN1	104	**0.799**	**0.690**	61	0.822	0.708	72	**0.796**	**0.676**
Low speed or low novelty													
0	0	0	0	LL1-3	764	0.954	0.946	498	0.964	0.956	659	0.948	0.938
1	0	0	0	LL2,3	104	0.955	0.932	61	0.959	0.934	72	0.955	0.930
0	1	0	0	LL1,3	79	0.939	0.909	80	0.956	0.935	34	0.942	0.905
0	0	1	0	LL1,2	174	0.939	0.904	85	0.954	0.917	157	0.936	0.898
0	0	0	1	LL1,2	243	0.923	0.891	217	0.939	0.910	176	0.924	0.890
1	0	1	0	LL2	33	0.949	0.894	41	0.952	0.896	39	0.946	0.889
0	1	1	0	LL1	51	0.925	0.852	36	0.934	0.861	32	0.926	0.842
1	0	0	1	LL2	57	0.911	0.834	59	0.930	0.870	51	0.905	0.823
0	0	1	1	LL1,2	249	0.889	0.826	218	0.905	0.847	243	0.875	0.804
0	1	0	1	LL1	26	0.905	0.816	35	0.917	0.847	20	0.918	0.826
1	1	0	0	LL3	315	0.821	0.743	280	0.875	0.822	309	0.856	0.794
1	0	1	1	LL2	127	0.850	0.714	132	0.870	0.759	125	0.838	**0.699**
0	1	1	1	LL1	82	0.832	**0.676**	106	0.843	0.718	77	0.847	0.702

Notes: For the configuration column, pai refers to attention partition by issues, pam refers to attention partition by modules, aui refers to attention augmentation by issues, and aum refers to attention augmentation by modules. Labels in the Main Result column presents simplified configurations that have include the corresponding configuration in the row. Values in bold are below the threshold of 0.80 for raw consistency and 0.7 for PRI consistency.

Table 4.8 Robustness Analyses of Collective-Attention Configurations (Robustness 4-6)

Configuration				Main result	Redefining primary focus (Robustness 4)			Feature-only analyses (Robustness 5)			Scoping involved developers (Robustness 6)		
					# of cases	Consistencies		# of cases	Consistencies		# of cases	Consistencies	
pai	pam	aui	aum	Label		Raw	PRI		Raw	PRI		Raw	PRI
High speed													
1	1	1	1	HS1	444	0.865	0.831	649	**0.798**	0.749	213	0.832	0.762
1	1	0	1	HS1	115	0.832	0.752	114	0.818	0.747	97	0.841	0.753
High novelty													
1	1	1	1	HN1	444	**0.799**	0.729	649	0.804	0.740	213	**0.794**	0.697
Low speed													
0	0	0	0	LS1-2	421	0.871	0.840	665	0.830	0.783	325	0.920	0.889
0	1	0	0	LS2	118	0.804	**0.693**	137	0.806	**0.679**	378	0.881	0.832
0	0	1	0	LS1	76	0.822	**0.660**	105	**0.797**	**0.635**	27	0.899	0.744
Low novelty													
0	0	0	0	LN1	421	**0.774**	0.728	665	**0.782**	0.736	325	0.831	0.780
1	0	0	0	LN1	46	0.806	0.701	94	**0.798**	**0.672**	102	0.845	0.762
Low speed or low novelty													
0	0	0	0	LL1-3	421	0.966	0.960	665	0.952	0.943	325	0.984	0.980
1	0	0	0	LL2,3	46	0.964	0.944	94	0.950	0.919	102	0.969	0.954
0	1	0	0	LL1,3	118	0.933	0.901	137	0.935	0.899	378	0.964	0.954
0	0	1	0	LL1,2	76	0.955	0.921	105	0.932	0.881	27	0.979	0.956
0	0	0	1	LL1,2	140	0.931	0.902	198	0.943	0.921	33	0.968	0.937
1	0	1	0	LL2	34	0.931	0.864	78	0.952	0.901	34	0.966	0.924
0	1	1	0	LL1	68	0.911	0.802	40	0.893	0.793	72	0.945	0.892
1	0	0	1	LL2	33	0.937	0.877	51	0.905	0.827	7	0.960	0.890
0	0	1	1	LL1,2	99	0.890	0.818	176	0.903	0.837	16	0.949	0.859
0	1	0	1	LL1	52	0.924	0.866	75	0.927	0.866	36	0.945	0.894
1	1	0	0	LL3	161	0.847	0.762	187	0.815	0.727	1128	**0.787**	0.730
1	0	1	1	LL2	90	0.861	0.735	133	0.856	0.741	7	0.945	0.824
0	1	1	1	LL1	94	0.831	**0.679**	130	0.814	**0.655**	35	0.903	0.777

Notes: For the configuration column, pai refers to attention partition by issues, pam refers to attention partition by modules, aui refers to attention augmentation by issues, and aum refers to attention augmentation by modules. Labels in the Main Result column presents simplified configurations that have include the corresponding configuration in the row. Values in bold are below the threshold of 0.80 for raw consistency and 0.70 for PRI consistency.

In **Robustness 2**, we adjusted our calibration strategy by varying the cross-over point 25% up and 25% down. As shown in Table 4.7, for the two speed outcomes (i.e., high speed, low speed), the solution configurations are all strongly supported. For high novelty, the solution configuration is strongly supported in the robustness test with cross-over point 25% up but moderately supported in the robustness test with cross-over point 25% down. For low novelty, the solution configurations are moderately supported in both tests. For the low-speed-or-low-novelty outcome, the solution configurations are strongly supported except one with PRI score slightly lower than 0.70 (i.e., 0.68 < 0.70). Overall, all solution configurations in the main results are supported in Robustness 2.

In **Robustness 3**, we changed our operationalization of module-oriented attention partition and attention augmentation by measuring the two constructs at the file level, rather than folder level. As shown in Table 4.7, for the two speed outcomes (i.e., high speed, low speed), the solution configurations are all strongly supported. For the two novelty outcomes (i.e., high novelty, low novelty), the solution configurations are moderately supported. For the low-speed-or-low-novelty outcome, the solution configurations are strongly supported. Overall, all solution configurations in the main results are supported in Robustness 3.

In **Robustness 4**, we adjusted our definition of primary focus by including issues (or modules) at not only the highest but also the second highest rank of innovative work. As shown in Table 4.8, for high speed, the solution configurations are strongly supported. For low speed and for two novelty outcomes (i.e., high novelty, low novelty), the solution configurations are moderately supported. For the low-speed-or-low-novelty outcome, the solution configurations are strongly supported except one with PRI score slightly lower than 0.70 (i.e., 0.68 < 0.70). Overall, all solution configurations in the main results are supported in Robustness 4.

In **Robustness 5**, we scoped to innovative work (or innovative actions) regarding only feature requests to measure our variables of interest, rather than innovative work regarding all types of issues, including feature requests, bug reports, and documentation updates. As shown in Table 4.8, for high speed and for high novelty, the solution configurations are strongly supported. For low speed and for low novelty, the solution configurations are moderately supported. For the low-speed-or-low-novelty outcome, the solution configurations are strongly supported except one with PRI score slightly lower than 0.70 (i.e., 0.66 < 0.70). Overall, all solution configurations in the main results are supported in Robustness 5.

In **Robustness 6**, we excluded individuals at the periphery of a focal collective, i.e., developers who made less than 15 lines of code change during a focal release cycle of an OSS project. As shown in Table 4.8, for two speed outcomes (i.e., high speed, low speed) and for low novelty, the solution configurations are strongly supported. For high novelty, the solution configuration is moderately supported (with raw consistency as 0.79). For the low-speed-or-low-novelty outcome, the solution configurations are strongly supported except one with raw consistency score slightly lower than 0.80 (i.e., 0.79 < 0.80). Overall, all solution configurations in the main results are supported in Robustness 6.

4.2 CONFIGURATIONS OF COLLECTIVE ATTENTION AND CONTINGENCIES FOR INNOVATION PRODUCTIVITY

Taking contingencies into consideration, we discover contingent collective-attention solutions for the dual goal (i.e., both high speed and high novelty) and the singular goals (i.e., high speed, high novelty) in this section. We first present the main results with the contingencies considered and then proceed to check the robustness of the findings.

4.2.1 Main Results

Table 4.9 illustrates configurations for all three innovation productivity goals when we incorporate contingent conditions. The contingencies pertain to the scope of innovative work (measured by the number of issues and the number of modules), availability of developers (measured by the number of developers), and temporal aspects of a focal release cycle (including project maturity and the length of release cycle).

For parsimony, we combined the number of issues and the number of modules into one measure, i.e., the scope of innovative work. Specifically, after obtaining the calibrated fuzzy-set measures for the two variables, we used the intersection of the two fuzzy sets to measure the scope of innovative work. The intersection was calculated by selecting the lower value of two fuzzy-set measures (i.e., using logical AND). For example, if a case’s membership in the fuzzy set of many issues is 0.95, and its membership in the fuzzy set of many modules is 0.10, then its membership to a broad scope of innovative work is 0.10. In other words, a broad scope of innovative work presents when there are both many issues and many modules. If there are many issues but few modules or few issues but many modules, the scope of innovative work will be limited. In our sample, 22% cases are in such unbalanced situations, whereas 39% cases are in the situation of few issues and few modules (or many issues and many modules).

We also conducted analyses without combining the number of issues and the number of modules. For the dual goal, we find that the overall coverage (or explanatory power) of solutions stays the same, and the configurations are exactly the same as HH1’ and HH2’ in Table 4.9, with a broad scope of innovative work expressed as the presence of both many issues and many modules. For the two singular goals, the overall coverages increase slightly (about 5%), but the solutions are much more complex (with the number of solutions doubled). Zooming into the truth tables, we find most solution configurations have both many issues and many modules, which is equivalent to a broad scope of innovative work. Besides, solution configurations with either few issues or few modules have very small numbers of cases (no more than 12), indicating that they are not very relevant. Thus, we choose to sacrifice potential additional insights from those detailed configurations to maintain parsimony of our results.

The final four contingent conditions and the four collective-attention constructs yield 256 possible configurations. With this enlarged solution space, we observe limited diversity in our sample and imposed a frequency threshold (or cutoff) of 16. Consequently, only configurations with the number of cases equal to or greater than 16 are considered as a candidate solution for the outcome. The selected configurations capture 77% cases in our sample. Prior literature has suggested that at least 75% to 80% cases in the sample should be captured after imposing a frequency threshold in fsQCA (Ragin 2018). We intentionally set the threshold a little higher to assure that the discovered solutions are relevant. However, if we further increase the frequency threshold as 17, the captured cases will drop to 74%. As to consistency thresholds for necessity and consistency analyses, we use the same ones that we use in the collective-attention configuration analyses (see Section 4.1). For necessity, the threshold is 0.90; for sufficiency, the threshold for raw consistency is 0.80, and the threshold for PRI score is 0.70 (it may be slightly lowered based on observations about gaps between PRI scores).

Based on these thresholds, we find two contingent solutions for the **dual goal of achieving both high speed and high novelty**. By saying *contingent solutions* (or configurations), we mean configurations of collective attention that are sufficient to elicit the outcome under specific contingent conditions regarding scope of innovative work, number of developers, project maturity, and length of release cycle. Recall that we did not find any solution configuration for the dual goal when we consider only collective-attention constructs. However, the analysis of solutions for the negation of the dual goal tells us that 13 of 16 possible configurations can be ruled out, because they lead to low speed or low novelty. It is interesting that the remaining three configurations are all covered in our contingent solutions (see HH1' and HH2' in Table

4.9), meaning that they can bring out both high speed and high novelty under some specific combinations of contingent conditions.

As shown in Table 4.9, the analytical results for the dual goal of both high speed and high novelty indicate that:

- If a focal release cycle that is short and is at the early stage of an OSS project with many developers working on a broad scope of innovative work, a combination of attention partition by both issues and modules and attention augmentation by issues is sufficient for achieving the dual goal (see HH1');
- Keeping all other contingencies as the same, if the focal release cycle is at the mature stage, attention augmentation by issues should be changed to attention augmentation by modules while attention partition by issues and by modules are still present (see HH2');
- Although some elements occur in both configurations, none of the elements are necessary (with consistency lower than 0.90).

Mapping to organizing modes, the contingent solutions indicate that an issue-oriented dual focus of collective attention (i.e., differentiating primary focus and overlapping secondary focus) benefits the achievement of the dual goal in the early stage of an OSS project; however, if the project is at a mature stage, the dual focus should be module-oriented. The dual focus of collective attention with two-way orientation works for both the early and the mature stages of an OSS project. If the dual focus of collective attention is present for one orientation (issue or module), the collective needs to only differentiate individuals' primary foci for the other orientation, meaning that attention augmentation for the other orientation is redundant (or can be either present or absent).

Table 4.9 Configurations of Collective Attention and Contingencies for Innovation Productivity

	Both high		**High speed**				**High novelty**		
Configuration	HH1’	HH2’	HS1’	HS2’	HS3’	HS4’	HN1’	HN2’	HN3’
Collective attention constructs									
Attention partition by issues	⬤	●	●	●	⬤	●	●	●	●
Attention partition by modules	●	●	●	⬤	●	⬤	●	●	
Attention augmentation by issues	⬤			⬤	⬤	⨂	⬤		●
Attention augmentation by modules		⬤				⬤		⬤	⬤
Contingent conditions									
Scope of innovative work	●	⬤	⬤	⬤		●	⬤	⬤	⬤
Number of developers	●	●	●	●	●	●	●	●	●
Project maturity	⨂	⬤		⬤	●				
Length of release cycle	⨂	⨂	⨂		⨂				●
Raw coverage	0.191	0.254	0.317	0.256	0.193	0.159	0.367	0.370	0.222
Unique coverage	0.097	0.160	0.095	0.064	0.019	0.025	0.053	0.057	0.017
Consistency	0.816	0.861	0.984	0.921	0.948	0.904	0.826	0.826	0.865
Overall solution coverage	0.351		0.443				0.441		
Overall solution consistency	0.829		0.912				0.804		
Frequency cutoff	16		16				16		
Consistency cutoff	0.823		0.862				0.832		

Note: ⬤ refers to present core; ● refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals. We add a single quotation mark after the labels of solution configurations with contingent conditions under consideration.

For the singular goal of high speed, the inclusion of contingent conditions helps us to see additional contingent solutions regarding collective attention. Those solutions are not covered by the solution configuration HS1 (in Table 4.5), discovered by considering only the four collective-attention constructs (i.e., a combination of attention partition by both issues and modules and attention augmentation by modules). Specifically, as shown in Table 4.9, the analytical results for high speed indicate that:

- If a focal release cycle is short and has many developers working on a broad scope of innovative work, a combination of attention partition by both issues and modules, but not attention augmentation by either issues or modules, can also elicit high speed (see HS1'[3]);

- A combination of attention partition by both issues and modules and attention augmentation by issues, but not attention augmentation by modules, can also elicit high speed,
    - if a focal release cycle is long and is at the mature stage of an OSS project with many developers working on a broad scope of innovative work, (see HS2'), or
    - if a focal release cycle is short and is at the mature stage of an OSS project with many developers working on a limited scope of innovative work (see HS3').

However, the inclusion of contingent conditions does not improve the explanatory power of resultant solutions, with the overall coverage as 0.44. Recall that the overall coverage of the solution (i.e., HS1) for considering only collective-attention constructs is 0.45. One reason is that we imposed a relatively high threshold for discovering solution configurations (0.862 for raw consistency and 0.75 for PRI consistency) with contingencies. We observe a salient gap regarding PRI consistency after 0.75 (i.e., the consistency directly drops to 0.71 after that). Prior literature stresses that instead of blindly implement the conventional benchmarks, a researcher needs to closely observe gaps surrounding such benchmarks and makes adjustments accordingly when establishing the consistency threshold (Ragin 2008; Ragin 2017). Thus, we slightly sacrificed coverage to obtain highly consistent solutions. The other reason is that HS1 has already covered a lot of information in HS1'-HS4' regarding collective attention allocation. Most cases (about 77%) that fall into configurations HS1'-HS4' have their collective-attention features resembling features identified by HS1. Configuration HS1 is probably the most representative solution for high speed.

[3] Note that to differentiate configurations with contingent conditions under consideration, we add a single quotation mark after their labels.

In addition to the representativeness, we find that configuration HS1 is contingency-robust. In Table 4.10, we zoom into the truth table and take a closer look at all configurations that have collective-attention part resembles features identified by HS1 and that have the number of cases greater than the frequency threshold 16. Note that almost all of them have consistency scores equal or above the conventional or recommended benchmarks (i.e., 0.80 for raw consistency and 0.70 for PRI consistency). For the only one that has raw consistency as 0.74 and PRI consistency as 0.41, its consistency is still the highest when compared with other configurations that have the same contingent conditions but different collective-attention features and that have no less than 16 cases. The highest raw consistency of these other configurations is 0.71, and the highest PRI consistency is 0.38. To conclude, the solution configuration for high speed (HS1), discovered by considering only collective-attention, is not only representative but also contingency-robust.

Turning to the singular goal of high novelty, we also find additional contingent solutions for attaining this goal when taking contingent conditions into consideration. Those additional contingent solutions reflect that some collective-attention conditions in the solution configuration HN1 (i.e., a combination of attention partition by both issues and modules and attention augmentation by both issues and modules) can be relaxed under specific contingent conditions. As shown in Table 4.9, the analytical results indicate for high novelty that:

- If a focal release cycle has many developers working on a broad scope of innovative work, attention augmentation by issues or by modules becomes a redundant element, meaning that one of them can be either present or absent when other three collective-attention elements are present (see HN1’ and HN2’);

- If the release cycle has an additional condition of being long, attention partition by modules can be relaxed but attention augmentation by both issues and modules and attention partition by issues need to be present (see HN3’).

Different from results for high speed, we find that the inclusion of contingent conditions noticeably improves the explanatory power of solution configurations (i.e., the overall coverage increases from 0.35 to 0.44). The collective-attention features identified by configuration HN1 is still a more representative solution than the additional contingent solutions, but the gap between their representativeness is much less. About 38% cases falling into HN1’-HN3’ have their collective attention resembling features identified by additional contingent solutions (rather than configuration HN1).

Besides, similar to results for high speed, we find that the solution configuration HN1, discovered by considering only collective-attention conditions, is a contingency-robust solution. As shown in Table 4.10, all configurations that have collective-attention part resembles features identified by HN1 and that have no less than 16 cases (i.e., the frequency cutoff) have raw consistency scores above the conventional threshold 0.8. Only two of them have PRI scores (i.e., 0.62 and 0.60) below the recommended threshold 0.70. For these two, their PRI scores are still the highest when compared with other configurations that have the same contingent conditions but different collective-attention features and that have no less than 16 cases. It means that HN1 is still a decent solution if a collective wants to achieve high novelty in those unfavorable situations. To conclude, configuration HN1 is a representative and contingency-robust solution for achieving high novelty, and the contingent solutions in HN1’-HN3’ are also relevant.

Table 4.10 Truth Table on Configurations with Collective-Attention Solutions for the Singular Goals and Contingent Conditions

Collective attention				Contingent conditions				Number of cases	Raw consistency	PRI consistency
pai	pam	aui	aum	$scope$	$numd$	pm	len			
Solution configuration for high speed (HS1): $pai * pam * aum$										
1	1	0	1	1	1	1	0	37	0.987	0.981
1	1	0	1	1	1	0	0	27	0.984	0.975
1	1	0	1	1	1	1	1	22	0.897	0.817
1	1	0	1	1	1	0	1	22	0.862	0.750
1	1	1	1	1	1	1	0	133	0.986	0.982
1	1	1	1	1	1	1	1	127	0.894	0.839
1	1	1	1	1	1	0	1	121	0.810	0.698
1	1	1	1	1	1	0	0	104	0.985	0.980
1	1	1	1	0	1	1	0	16	0.932	0.853
1	1	1	1	0	1	0	1	16	0.744	**0.407**
Solution configuration for high novelty (HN1): $pai * pam * aui * aum$										
1	1	1	1	1	1	1	0	133	0.884	0.822
1	1	1	1	1	1	1	1	127	0.895	0.828
1	1	1	1	1	1	0	1	121	0.898	0.828
1	1	1	1	1	1	0	0	104	0.832	0.730
1	1	1	1	0	1	0	1	16	0.832	**0.620**
1	1	1	1	0	1	1	0	16	0.805	**0.602**

Note: For the collective attention, pai refers to attention partition by issues, pam refers to attention partition by modules, aui refers to attention augmentation by issues, aum refers to attention augmentation by modules. For the contingent conditions, $scope$ refers to scope of innovative work, $numd$ refers to the number of involved developers, pm refers to project maturity, and len refers to the length of release cycle. The bold values are below the threshold of 0.8 for raw consistency and 0.70 for PRI consistency.

4.2.2 Robustness Tests

We first check the robustness of solution configurations for the dual goal and then the robustness of solution configurations for the singular goals with a series of robustness tests.

Robustness Tests on Solution Configurations for the Dual Goal

We conducted six blocks of robustness analyses on configurations for the dual goal of both high speed and high novelty, i.e., HH1’ and HH2’ in Table 4.9. Overall, the results show that the two configurations are robust solutions for achieving the dual goal.

Table 4.11 Tobit Models of Contingent Solution-Configuration Deviation on the Dual Goal Regarding Speed and Novelty (Robustness 1)

Independent variable	**Membership in both high speed and high novelty**		
	Model 1	**Model 2**	**Model 3**
Ideal types fit (for HH1’-2’)	0.391*** (0.021)		
HH1’ deviation		-0.394*** (0.026)	
HH2’ deviation			-0.398*** (0.019)
Constant	0.882*** (0.039)	0.946*** (0.054)	0.945*** (0.039)
F score	360.01***	221.37***	446.09***
Pseudo R^2	0.6796	0.5430	0.6465
Observations	3,052	3,052	3,052

Note: We use the two-limit Tobit regression model and allow for intragroup correlation within the same open-source software project (or repository). Robust standard errors are reported in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Table 4.11 shows the results for **Robustness 1**, i.e., the deviation score analyses. Specifically, Model 1 shows the results of a collective’s overall ideal types fit across two configurations. The fit coefficient is positive and significant, which indicates that a collective is more likely to achieve the dual goal if its collective-attention characteristics and its contingent conditions resemble features identified by any of the two ideal types, i.e., configurations HH1’ and HH2’. Model 2 and Model 3 show the results of a collective’s deviation from individual solution configurations. The coefficients are negative and significant, indicating that closer distance to HH1’ (or HH2’) is associated with the outcome of having both high speed and high novelty. What’s more, the explanatory power of each configuration and the two configurations overall is very high, with pseudo R-squares ranging from 0.54 to 0.68.

Table 4.12 shows results for **Robustness 2-6**, i.e., five additional analyses using the set-theoretic approach, which tweak decisions in the main analyses regarding calibration anchors (Robustness 2), operationalization of module-oriented collective attention constructs

(Robustness 3), definition of primary focus (Robustness 4), focal types of issue (Robustness 5), and scope of involved developers (Robustness 6). The raw consistency scores of four solution configurations covered in configuration HH1’ and HH2’ are above the conventional benchmark 0.8 across all robustness analysis. However, the PRI scores of some configurations are below the established threshold 0.70 (but still all above 0.60), meaning that the configurations are at least weakly supported.

Table 4.12 Robustness Analyses of Contingent Configurations for the Dual Goal: Both High Speed and High Novelty (Robustness 2-6)

Configuration								Main result	Sensitivity analyses (Robustness 2)					
Collective attention				Contingencies					Cross-over 25% up			Cross-over 25% down		
									# of	Consistencies		# of	Consistencies	
pai	pam	aui	aum	$scope$	$numd$	pm	len	Label	cases	Raw	PRI	Cases	Raw	PRI
1	1	1	1	1	1	0	0	HH1’	75	0.848	0.716	104	0.812	**0.668**
1	1	1	0	1	1	0	0	HH1’	34	0.879	0.718	14	0.868	**0.654**
1	1	1	1	1	1	1	0	HH2’	81	0.897	0.811	168	0.852	0.759
1	1	0	1	1	1	1	0	HH2’	45	0.881	0.755	43	0.846	**0.685**
									File-level analysis (Robustness 3)			**Redefining primary focus (Robustness 4)**		
									# of	Consistencies		# of	Consistencies	
pai	pam	aui	aum	$scope$	$numd$	pm	len	Label	cases	Raw	PRI	Cases	Raw	PRI
1	1	1	1	1	1	0	0	HH1’	106	0.830	**0.692**	118	0.830	0.700
1	1	1	0	1	1	0	0	HH1’	19	0.888	**0.698**	25	0.860	**0.672**
1	1	1	1	1	1	1	0	HH2’	142	0.880	0.795	146	0.879	0.796
1	1	0	1	1	1	1	0	HH2’	35	0.861	0.700	13	0.855	**0.641**
									Feature-only analyses (Robustness 5)			**Scoping involved developers (Robustness 6)**		
									# of	Consistencies		# of	Consistencies	
pai	pam	aui	aum	$scope$	$numd$	pm	len	Label	cases	Raw	PRI	Cases	Raw	PRI
1	1	1	1	1	1	0	0	HH1’	61	0.849	0.737	37	0.842	**0.662**
1	1	1	0	1	1	0	0	HH1’	21	0.822	**0.631**	61	0.821	**0.629**
1	1	1	1	1	1	1	0	HH2’	98	0.869	0.795	26	0.892	0.763
1	1	0	1	1	1	1	0	HH2	29	0.845	0.704	9	0.884	0.703

Notes: The configurations are sorted by contingent conditions. Thus, configurations with the same contingency are grouped together; collective-attention conditions combined with different contingencies are divided by grey-color zones. For the configuration column, pai refers to attention partition by issues, pam refers to attention partition by modules, aui refers to attention augmentation by issues, aum refers to attention augmentation by modules, $scope$ refers to scope of innovative work, $numd$ refers to the number of involved developers, pm refers to project maturity, and len refers to the length of release cycle. Labels in the Main Result column presents simplified configurations that have include the corresponding configuration in the row. The bold values are 1) below the threshold of 0.80 for raw consistency or 0.70 for PRI consistency or 2) below the frequency threshold of 16.

Table 4.13 Comparing the Main Results and Robustness 4-6 Results for the Dual Goal

	Main results		Robust. 4	Robust. 5	Robust. 6		
Configuration	HH1’	HH2’	HH1’_4	HH1’_5	HH1’_6	HH2’_6	HH3’_6
Collective attention constructs							
Attention partition by issues	⬤	●	⬤	●	●	●	●
Attention partition by modules	●	●	●	⬤	●	●	●
Attention augmentation by issues	⬤		⬤		●	⬤	●
Attention augmentation by modules		⬤		⬤	⬤		⬤
Contingent conditions							
Scope of innovative work	●	⬤	●	●	●	⬤	●
Number of developers	●	●	●	●	●	●	●
Project maturity	⨂	⬤				⬤	⬤
Length of release cycle	⨂	⨂	⨂	⨂	⨂	⨂	
Raw coverage	0.191	0.254	0.365	0.346	0.148	0.229	0.146
Unique coverage	0.097	0.160	0.365	0.346	0.040	0.121	0.038
Consistency	0.816	0.861	0.827	0.820	0.856	0.871	0.844
Overall solution coverage	0.351		0.365	0.346	0.307		
Overall solution consistency	0.829		0.827	0.820	0.836		
Frequency cutoff	16		16	16	19*		
Consistency cutoff	0.823		0.830	0.828	0.842		

Note: * We use the same frequency threshold 16 for Robustness 6, but coincidentally, no configurations have the number of cases as 16, 17, or 18; the actual frequency threshold turns to 19. Note that for all above analyses, we take the first four best performing configurations; their raw consistency scores are all above 0.8, and their PRI scores are 0.65 or above. Robust. 4 refers to the analyses that redefine primary focus by including also the second highest rank of innovative work. Robust. 5 refers to the analyses that scope to innovative work on feature requests. Robust. 6 refers to the analyses that exclude developers at the periphery (who have made less than 15 lines of code change) from a collective. ⬤ refers to present core; ● refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals.

Consider the results of **Robustness 2-3**. Although some configurations have PRI scores slightly lower (above 0.65) than our established threshold 0.70, the four solution configurations covered in HH1’ and HH2’ are still clearly the best-performing ones. In other words, the contradiction is trivial, and the pattern does not change. However, the other three blocks of robustness analyses **Robustness 4-6** are in a tricky situation. In their truth tables, some configurations have better consistency performance (higher PRI scores) than the four covered in HH1' and HH2', meaning that the pattern of the solution configurations in these robustness tests would be different from the solution configurations in the main results. Table 4.13 presents a contrast between solution configurations in the main results and in these robustness tests.

- In Robustness 4, we redefined primary focus by including the second highest rank of innovative work and observed that no matter whether a focal release cycle is at the early or mature stage of an OSS project, attention augmentation by modules was redundant for achieving the dual goal (see HH1'_4);
- In Robustness 5, we considered only innovative work on feature requests and observed that no matter whether a focal release cycle is at the early or mature stage of an OSS project, attention augmentation by issues was redundant for achieving the dual goal (see HH1'_5);
- In Robustness 6, we excluded developers at the periphery who made less than 15 lines of code change and observed that at the mature stage of an OSS project,

These three observations contradict our finding in the main result that at the early stage of an OSS project, attention augmentation by modules is redundant for achieving the dual goal, whereas at the mature stage, attention augmentation by issues is redundant. Therefore, in the main results, our finding on the redundancy of attention augmentation by modules (by issues) at the early (the mature) stage of an OSS project is only partially supported (not supported by Robustness 4-6).

Besides, our finding in the main results that for a release cycle that is short and has many developers working on a broad scope of innovative work, a dual focus of collective attention (attention partition and attention augmentation) with two-way orientation (issue and module) achieves the dual goal is supported across all robustness tests. Moreover, configuration HH3'_6 in Robustness 6 indicates an additional situation (not occurred in the main results) in which the dual focus of collective attention with two-way orientations elicits the dual goal. The situation refers to a release cycle that is long and is at the mature stage of an OSS project with many developers working on a broad scope of innovative work.

Robustness Tests on Solution Configurations for the Singular Goals

We also did six blocks of robustness analyses on configurations for the singular goals.

Robustness 1: Deviation score analyses. Table 4.14 presents the results for **high speed**. Model 1 shows the results of ideal types fit. The fit coefficient is positive and significant, indicating that a collective is more likely to achieve high speed if its collective attention and contingent conditions resemble features identified by any of the four ideal types, i.e., configurations HS1'-HS4'. Model 1-5 shows the results of a collective's deviation from individual configurations. The coefficients are negative and significant, indicating that closer distance to each configuration is associated with the outcome of high speed. The explanatory power of each configuration and the configurations overall is reasonably high, with lowest pseudo R-square as 0.32. Table 4.15 presents results for **high novelty**. Similarly, the coefficient for ideal types fit (in Model 1) is positive and significant, and the coefficient for deviation from each configuration (HN1', HN2' or HN3' in Model 2, 3 or 4) is negative and significant. The explanatory power of each configuration and the configurations overall is reasonable, with pseudo R-squares ranging from 0.13 to 0.14. Therefore, the deviation score analyses indicate that solution configurations in the main results for the singular goals are robust.

Table 4.14 Tobit Models of Contingent Solution-Configuration Deviation on Speed (Robustness 1)

Independent variable	Membership in high speed				
	Model 1	**Model 2**	**Model 3**	**Model 4**	**Model 5**
Ideal types fit (for HS1'-4')	0.468*** (0.021)				
HS1' deviation		-0.524*** (0.019)			
HS2' deviation			-0.412*** (0.027)		
HS3' deviation				-0.530*** (0.029)	
HS4' deviation					-0.498*** (0.033)
Constant	1.081*** (0.033)	1.202*** (0.030)	1.110*** (0.048)	1.285*** (0.051)	1.318*** (0.063)
F score	476.58***	755.02***	241.60***	345.29***	232.67***
Pseudo R^2	0.4766	0.5309	0.3396	0.3868	0.3157
Observations	3,052	3,052	3,052	3,052	3,052

Note: We use the two-limit Tobit regression model and allow for intragroup correlation within the same open-source software project (or repository). Robust standard errors are reported in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Table 4.15 Tobit Models of Contingent Solution-Configuration Deviation on Novelty (Robustness 1)

Independent variable	Membership in high novelty			
	Model 1	**Model 2**	**Model 3**	**Model 4**
Ideal types fit (for HN1'-3')	0.259*** (0.019)			
HN1' deviation		-0.264*** (0.020)		
HN2' deviation			-0.264*** (0.020)	
HN3' deviation				-0.292*** (0.022)
Constant	0.814*** (0.030)	0.845*** (0.032)	0.842*** (0.032)	0.941*** (0.041)
F score	183.73***	180.66***	181.39***	168.92***
Pseudo R^2	0.1399	0.1364	0.1347	0.1308
Observations	3,052	3,052	3,052	3,052

Note: We use the two-limit Tobit regression model and allow for intragroup correlation within the same open-source software project (or repository). Robust standard errors are reported in parentheses. *** p<0.01, ** p<0.05, * p<0.1

Table 4.16 Summary of Robustness Analyses on Contingent Configurations for a Singular Goals: High Speed or High Novelty (Robustness 2-6)

Configuration (not simplified)								Main	Robustness Statistics					
Collective attention				Contingencies				Result	# of cases		Raw consist.		PRI consist.	
pai	pam	aui	aum	$scope$	$numd$	pm	len	Label	Mean	Min	Mean	Min	Mean	Min
High speed														
1	1	1	1	0	1	1	0	HS3’	**12.50**	**7**	0.942	0.923	0.872	0.842
1	1	1	0	0	1	1	0	HS3’	16.50	**4**	0.912	0.897	0.787	0.749
1	1	1	1	1	1	0	0	HS1’	83.50	37	0.985	0.976	0.980	0.967
1	1	0	1	1	1	0	0	HS1’	25.17	**11**	0.980	0.969	0.971	0.955
1	1	1	0	1	1	0	0	HS1’	29.00	**14**	0.974	0.961	0.956	0.937
1	1	0	0	1	1	0	0	HS1’	23.33	**6**	0.967	0.955	0.942	0.920
1	1	0	1	1	1	0	1	HS4’	18.33	**2**	0.865	0.847	0.747	0.713
1	1	1	1	1	1	1	0	HS1’-3’	110.17	26	0.984	0.968	0.979	0.958
1	1	1	0	1	1	1	0	HS1’-3’	40.00	24	0.979	0.966	0.966	0.948
1	1	0	1	1	1	1	0	HS1’,4’	29.00	**9**	0.982	0.961	0.974	0.946
1	1	0	0	1	1	1	0	HS1’	37.83	**11**	0.974	0.954	0.957	0.927
1	1	1	1	1	1	1	1	HS2’	117.67	35	0.888	0.854	0.829	0.789
1	1	0	1	1	1	1	1	HS4’	19.83	**8**	0.894	0.870	0.807	0.767
1	1	1	0	1	1	1	1	HS2’	41.33	18	0.882	0.864	0.779	0.733
High novelty														
1	1	1	0	1	1	0	0	HN1’	29.00	**14**	0.868	0.828	0.745	**0.692**
1	1	1	1	1	1	0	0	HN1’,2’	83.50	37	0.844	0.820	0.744	0.716
1	1	0	1	1	1	0	0	HN2	25.17	**11**	0.859	0.843	0.721	**0.688**
1	1	1	1	1	1	0	1	HN1’-3’	106.67	36	0.881	0.848	0.798	0.759
1	1	1	0	1	1	0	1	HN1’	46.00	22	0.883	0.864	0.759	0.739
1	1	0	1	1	1	0	1	HN2’	18.33	**2**	0.875	0.854	0.729	**0.695**
1	0	1	1	1	1	0	1	HN3’	**14.00**	**1**	0.902	0.879	0.739	**0.688**
1	1	1	1	1	1	1	0	HN1’,2’	110.17	26	0.888	0.862	0.825	0.794
1	1	1	0	1	1	1	0	HN1’	40.00	24	0.862	0.833	0.746	0.703
1	1	0	1	1	1	1	0	HN2’	29.00	**9**	0.874	0.856	0.764	0.732
1	1	1	1	1	1	1	1	HN1’-3’	117.67	35	0.888	0.862	0.813	0.787
1	1	1	0	1	1	1	1	HN1’	41.33	18	0.876	0.858	0.747	0.716
1	1	0	1	1	1	1	1	HN2’	19.83	**8**	0.902	0.878	0.798	0.768
1	0	1	1	1	1	1	1	HN3’	**14.17**	**1**	0.910	0.874	0.757	**0.693**

Notes: The configurations are sorted by contingent conditions. Thus, configurations with the same contingency are grouped together; collective-attention conditions combined with different contingencies are divided by grey-color zones. For the configuration column, pai refers to attention partition by issues, pam refers to attention partition by modules, aui refers to attention augmentation by issues, aum refers to attention augmentation by modules, $scope$ refers to scope of innovative work, $numd$ refers to the number of involved developers, pm refers to project maturity, and len refers to the length of release cycle. Labels in the Main Result column presents simplified configurations that have include the corresponding configuration in the row. The bold values are 1) below the threshold of 0.80 for raw consistency or 0.70 for PRI consistency or 2) below the frequency threshold of 16.

Robustness 2-6: Robustness tests regarding key decisions in the main analyses. Table 4.16 shows results of the other five blocks of robustness analyses, which used the set-theoretic approach and tweaked decisions in the main analyses. For **high speed** and for **high novelty**, as the distribution of cases over possible configurations changes, we observe changes in their solutions. Some configurations in the main results do not pass the frequency threshold in a robustness test and are considered counterfactual. Some additional configurations also appear. However, since these configurations all have limited numbers of cases, they are not as relevant to our core interpretations on the patterns of solution configurations. Besides, the sufficiency of solution configurations covered in the main results (configurations HS1'-HS4' and HN1'-HN3') hold across all robustness tests. Their raw consistency scores and PRI scores are all above the established thresholds (if slightly below a threshold, the gap is less than 0.02). These configurations are also clearly the best-performing ones in the truth tables. None of these observations rebut the robustness of the solutions. Thus, overall, the solution configurations for the singular goals are supported.

4.3 INCLUSION OF ADDITIONAL CONTINGENCIES

In this section, we present results for explorative analyses that include additional contingent conditions related to collective composition. The purpose is to check whether the inclusion of an additional contingency improves our results in terms of empirical relevance (higher coverage being more desirable), parsimony (fewer, parsimonious configurations being more optimal than a large number of configurations), and sense-making (more fundamental insights on contingent solutions of collective attention being preferred).

4.3.1 Collective Diversity as a Contingency

We considered an additional contingency on collective diversity. Prior literature has found that team diversity affects team production (Horwitz and Horwitz 2007; Ancona and Caldwell 1992) and group diversity affects group productivity in online open collaboration (Ren et al. 2016). To explore contingent solutions regarding diversity, we added a contingent condition: *tenure diversity*, i.e., the variability of individuals' experience on conducting innovative work (or coding) for a project. However, we did not consider interest diversity, another type of diversity that Ren et al. (2016) has studied in the open-innovation context. The reason is that we scope our empirical setting to projects on a specific topic, machine learning. The value of further differentiating developers' interest in subtopics of machine learning is very limited. Besides, adding two elements into the framework would increase the number of possible configurations from 256 to 1,024, which is huge. For parsimony, it is better not to include an additional diversity variable that is not valuable.

We took two steps to measure tenure diversity. First, we gauged individual developers' project tenure by counting days that had elapsed from the date when a developer first made a commit for a focal project to the date when a given release cycle of the project started. Second, we computed the coefficient of variation (CV) for individuals' project tenure. With each individual's project tenure denoted as T_i and the mean over m individuals' project tenure denoted as T_{mean}, the coefficient was calculated by using the formula: $[\sum_{i=1}^{m}(T_i - T_{mean})^2/n]^{1/2}/T_{mean}$. Similar to other variables, we converted the CV measure into a fuzzy-set measure by using a sample-dependent calibration strategy, i.e., setting its 75th percentile as full membership, 25th percentile as full non-membership, and 50th percentile as the cross-over point with most ambiguity.

Tables 4.17-4.19 shows configurational results for three productivity goals after adding tenure diversity. Since the possible configurations double after adding one condition, we lowered the frequency cutoff to 7, which allowed the selected configurations to capture about 77% (same as the proportion in main analyses) cases in our sample. **For the dual goal** of both high speed and high novelty, the overall coverage is only slightly improved (from 0.35 to 0.37) after adding tenure diversity, and we do not observe more parsimonious configurational results. Besides, the contingent solutions of collective attention are the same as what we have in the main results, although we observe that their effects may vary with the change of tenure diversity. **For the singular goals**, the overall coverage is either slightly or not improved (i.e., changed from 0.44 to 0.48 for high speed and from 0.443 to 0.435 for high novelty). The resultant configurations are much more complex, which greatly reduced the interpretability. Thus, we do not take tenure diversity into consideration in our main analyses.

Table 4.17 Configurations for the Dual Goal When Considering Tenure Diversity

	Both high speed and high novelty			
Configuration	HH1	HH2	HH3	HH4
Collective attention constructs				
Attention partition by issues (pai)	●	●	•	•
Attention partition by modules (pam)	●	•	•	●
Attention augmentation by issues (aui)	●	●		
Attention augmentation by modules (aum)	•	⨂	●	•
Contingent conditions included in theoretical framework				
Scope of innovative work ($scope$)	●	•	●	●
Number of developers ($numd$)	•	•	•	•
Project maturity (pm)		⨂	●	●
Length of release cycle (len)	⨂	⨂	⨂	
Additional contingency				
Tenure diversity (td)	⨂	•		⨂
Raw coverage	0.202	0.082	0.254	0.199
Unique coverage	0.050	0.021	0.074	0.034
Consistency	0.867	0.890	0.861	0.865
Overall solution coverage	0.368			
Overall solution consistency	0.835			
Frequency cutoff	7			
Consistency cutoff	0.841			

Note: ● refers to present core; • refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals.

Table 4.18 Configurations for High Speed When Considering Tenure Diversity

Configuration	**High Speed**									
	HS1a	HS1b	HS1c	HS2	HS3	H3b	HS3c	HS4	HS5	HS6
Collective attention constructs										
Attention partition by issues (pai)	●		⊗	●	⬤	⬤	⬤	⬤	●	●
Attention partition by modules (pam)	●	●	⊗	⬤	⊗	●	●	⊗	●	
Attention augmentation by issues (aui)		●	●	⬤	⬤	⬤	⬤	⊗	⨂	●
Attention augmentation by modules (aum)		●	●		●	●		⬤	⬤	●
Contingent conditions included in theoretical framework										
Scope of innovative work ($scope$)	⬤	⬤	⬤	⬤	●			⨂	⬤	●
Number of developers ($numd$)	●	●	⊗	●		●	●	⊗	●	⬤
Project maturity (pm)		⊗	⊗		⊗	●	●	⨂		⬤
Length of release cycle (len)	⨂	⨂	⨂		⨂	⨂	⨂	⊗		●
Additional contingency										
Tenure diversity (td)	●	⊗	⊗	⬤	⊗		●	⨂		⨂
Raw coverage	0.207	0.102	0.048	0.277	0.056	0.164	0.150	0.037	0.159	0.076
Unique coverage	0.016	0.028	0.010	0.076	0.009	0.018	0.007	0.007	0.037	0.010
Consistency	0.982	0.988	0.985	0.883	0.965	0.965	0.908	0.908	0.904	0.934
Overall solution coverage	0.484									
Overall solution consistency	0.887									
Frequency cutoff	7									
Consistency cutoff	0.846									

Note: ⬤ refers to present core; ● refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals.

Table 4.19 Configurations for High Novelty When Considering Tenure Diversity

Configuration	High Novelty						
	HN1a	HN1b	HN2	HN3	HN4a	HN4b	HN4c
Collective attention constructs							
Attention partition by issues (pai)	●			●	●	●	⊗
Attention partition by modules (pam)	⬤	⬤	⬤	⬤	●		⊗
Attention augmentation by issues (aui)	⬤	⬤				●	●
Attention augmentation by modules (aum)		●	⬤	⬤		●	●
Contingent conditions included in theoretical framework							
Scope of innovative work ($scope$)	⬤	⬤	⬤	⬤	⬤	⬤	⬤
Number of developers ($numd$)	●	●		●	●	●	⊗
Project maturity (pm)		⊗	⬤		⊗		⊗
Length of release cycle (len)		⊗			⬤	⬤	⬤
Additional contingency							
Tenure diversity (td)	●	⊗		⬤	⨂	⨂	⨂
Raw coverage	0.260	0.088	0.244	0.252	0.096	0.117	0.047
Unique coverage	0.037	0.023	0.033	0.008	0.014	0.007	0.011
Consistency	0.829	0.850	0.860	0.842	0.874	0.913	0.894
Overall solution coverage	0.435						
Overall solution consistency	0.815						
Frequency cutoff	7						
Consistency cutoff	0.846						

Note: ⬤ refers to present core; ● refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals.

4.3.2 Star Contributors as a Contingency

We considered another additional contingency of star contributors. Prior literature has suggested that the presence of star contributors affects outcomes of collective work (Aguinis and O'Boyle 2014) as the collective work includes increased complexity, reduced situational constraints (such as geographic distances and inability to access information) and flexible hierarchies. Open-source software development resembles such features. Thus, we added a contingent condition: *star contributors*, i.e., the extent to which innovative work in a focal release cycle of an OSS project is conducted by few developers. We measured it by identifying commits (i.e., innovative work) made within a given release cycle and then calculating the Herfindahl-Hirschman Index (HHI) of innovative work over individual developers. The same sample-dependent calibration strategy was used to calibrate this variable. Specifically, we set its 75th percentile as full membership, 25th percentile as full non-membership, and 50th percentile as the cross-over point with most ambiguity.

Tables 4.20-4.22 present the resultant configurations for achieving the three productivity goals when considering star contributors as a continency. Similarly, we set the frequency cutoff as 8, which allowed the selected configurations to capture about 77% cases in our sample. **For the dual goal** of both high speed and high novelty, the overall coverage is not improved by including star contributors (i.e., changed from 0.35 to 0.34). Besides, two configurations in the results are almost exactly the same as those in our main results, except that they have an additional contingent condition—the absence of star contributors—and that with this contingent condition added, a combination of attention partition by both issues and modules and attention augmentation by issues is sufficient for achieving the dual goal no matter whether a focal release cycle is at the early or mature project stage of an OSS project (while in our main results, this combination works only for release cycles at the early stage of an OSS project). **For the singular goals**, although the overall coverage is improved (about 8% for high speed and about 2% for high novelty), the resultant configurations are too complex to interpret. Thus, we also do not take star contributors into consideration in our main analyses.

Table 4.20 Configurations for the Dual Goal When Considering Star Contributors

Configuration	**Both high speed and high novelty**	
	HH1	HH2
Collective attention constructs		
Attention partition by issues	⬤	⬤
Attention partition by modules	⬤	●
Attention augmentation by issues	⬤	
Attention augmentation by modules		⬤
Contingent conditions included in theoretical framework		
Scope of innovative work	⬤	⬤
Number of developers	●	●
Project maturity		⬤
Length of release cycle	⨂	⨂
Additional contingency		
Star contributors ($star$)	⨂	⨂
Raw coverage	0.309	0.234
Unique coverage	0.105	0.030
Consistency	0.835	0.869
Overall solution coverage	0.339	
Overall solution consistency	0.830	
Frequency cutoff	8	
Consistency cutoff	0.828	

Note: ⬤ refers to present core; ● refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals.

Table 4.21 Configurations for High Speed When Considering Star Contributors

	High speed											
Configuration	HS1a	HS1b	HS1c	HS2	HS3a	HS3b	HS3c	HS4	HS5a	HS5b	HS6a	HS6a
Collective attention constructs												
Attention partition by issues (pai)	●	●		●	⬤	⬤	⬤	⬤	⬤	⬤	●	●
Attention partition by modules (pam)	●	●	●	●	●	●	⊗	⬤	●	⊗	●	
Attention augmentation by issues (aui)		●	●			●	⊗		⬤	⬤	●	●
Attention augmentation by modules (aum)		●	●		⊗		●	⬤	⊗	●	⬤	⬤
Contingent conditions included in theoretical framework												
Scope of innovative work $(scope)$	⬤	⬤	⬤	⬤			⊗	⬤			⬤	⬤
Number of developers $(numd)$	●	●	●	●	●	●	⊗	●	●	●	●	●
Project maturity (pm)			●	⬤	⬤	⬤	⬤		●	⊗	⬤	⬤
Length of release cycle (len)	⨂	⨂	⨂		⨂	⨂	⨂		⨂	⨂		●
Additional contingency												
Star contributors $(star)$	⊗		⊗	⨂	⨂	⨂	⨂	⨂		⊗		⊗
Raw coverage	0.278	0.216	0.149	0.285	0.099	0.169	0.036	0.360	0.094	0.060	0.220	0.135
Unique coverage	0.017	0.005	0.009	0.023	0.006	0.005	0.008	0.050	0.010	0.012	0.006	0.007
Consistency	0.985	0.986	0.984	0.913	0.916	0.956	0.913	0.906	0.932	0.954	0.929	0.895
Overall solution coverage	0.520											
Overall solution consistency	0.884											
Frequency cutoff	8											
Consistency cutoff	0.839											

Note: ⬤ refers to present core; ● refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals.

Table 4.22 Configurations for High Novelty When Considering Star Contributors

Configuration	High novelty								
	HN1a	**HN1b**	**HN1c**	**HN1d**	**HN2a**	**HN2b**	**HN2c**	**HN3**	**HN4**
Collective attention constructs									
Attention partition by issues	●	●			●	●	⊗	●	●
Attention partition by modules	⬤	⬤	⬤	⬤	●		⊗	⬤	⬤
Attention augmentation by issues	●		●	●	⬤	⬤	⬤	⬤	
Attention augmentation by modules	⬤	⬤	⬤	⬤		●	●		
Contingent conditions included in theoretical framework									
Scope of innovative work	⬤	⬤	⬤	⬤	⬤	⬤	⬤	⬤	⬤
Number of developers	●	●	●	●	●	●	⊗	●	●
Project maturity			●	⊗			⊗		⨂
Length of release cycle			⊗	●	⬤	⬤	⬤		
Additional contingency									
Star contributors ($star$)		⊗	⊗	●		⊗	●	⨂	⨂
Raw coverage	0.313	0.332	0.135	0.118	0.237	0.198	0.048	0.324	0.214
Unique coverage	0.008	0.027	0.006	0.003	0.009	0.012	0.011	0.012	0.012
Consistency	0.851	0.834	0.892	0.893	0.858	0.867	0.904	0.834	0.802
Overall solution coverage	0.462								
Overall solution consistency	0.806								
Frequency cutoff	8								
Consistency cutoff	0.837								

Note: ⬤ refers to present core; ● refers to present peripherals; ⨂ refers to absent core; ⊗ refers to absent peripherals.

5. DISCUSSION

In this chapter, we first interpret the configurational results and develop theoretical propositions that illuminate the relationship between collective attention allocation and innovation productivity in a release cycle of an open-source software (OSS) project (see Section 5.1). Then, we discuss the theoretical implications of our study (see Section 5.2). Finally, we discuss the practical implications of our study (see section 5.3) as well as its limitations and avenues for future research (see Section 5.4).

5.1 EFFECTIVE CONFIGURATIONS OF COLLECTIVE ATTENTION FOR INNOVATION PRODUCTIVITY IN OPEN-SOURCE SOFTWARE PROJECTS

Based on configurational results in Chapter 4, we develop propositions that illuminate effective configurations of collective attention for achieving three productivity goals, i.e., high speed, high novelty, and both. As shown in Table 5.1, we find support for the viewpoint that a collective-attention configuration, with the presence of all collective attention constructs (i.e., attention partition and attention augmentation by both issues and modules), is important for innovation productivity. It is a contingency-robust solution for the attainment of the singular goals (see HS1 and HN1 in Table 5.1); it also appears in configurational solutions that include contingencies (see HS1’-HS3’ and HN1’-HN3’ in Table 5.1); all these solution configurations are supported by a range of robustness tests.

Besides, the collective-attention configuration with the presence of all four characteristics (i.e., attention partition and attention augmentation by issues and by modules) is theoretically important as it indicates the most comprehensive configuration of collective attention, where a collective employs a **dual** focus of collective attention (attention partition and attention augmentation) with **two-way** orientation (issue and module).

We proceed by first interpreting the relationship between this comprehensive collective-attention configuration and each of the three productivity goals (see Section 5.1.1) and then explaining how conditions in the comprehensive attention configuration can be relaxed given the collective's innovation productivity goal and contingencies (see Section 5.1.2).

Table 5.1 Summary of Collective-Attention Solution Configurations for the Three Innovation Productivity Goals

Solution in Main Analyses	Contingency	Collective Attention		Evidence from robustness tests						
		Input orientation	Output orientation	1. Deviation score analyses	2. Sensitivity analyses		3. File-level analyses	4. Redefining primary analyses	5. Feature-only analyses	6. Scoping developer analyses
					Crossover up 25%	Crossover down 25%				
Collective attention → High speed (Overall solution coverage: 0.459)										
HS1	Robust	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Moderate	Strong
		$pa * \sim au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
Collective attention and contingencies → High speed (Overall solution coverage: 0.443)										
HS1'	$scope * numd * \sim len$	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * \sim au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * au$	$pa * \sim au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * \sim au$	$pa * \sim au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
HS2'	$scope * numd * pm$	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * au$	$pa * \sim au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
HS3'	$numd * pm * \sim len$	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * au$	$pa * \sim au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
HS4'	$scope * numd$	$pa * \sim au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
Collective attention → High novelty (Overall solution coverage: 0.347)										
HN1	Robust	$pa * au$	$pa * au$	Strong	Strong	Moderate	Moderate	Moderate	Strong	Moderate
Collective attention and contingencies → High novelty (Overall solution coverage: 0.441)										
HN1'	$scope * numd$	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * au$	$pa * \sim au$	Strong	Strong	Strong	Strong	Strong	Strong	Moderate
HN2'	$scope * numd$	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * \sim au$	$pa * au$	Strong	Strong	Moderate	Strong	Moderate	Strong	Strong
HN3'	$scope * numd * len$	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * au$	$\sim pa * au$	Strong	Moderate	Moderate	Strong	Strong	Moderate	Strong
Collective attention and contingencies → Both high speed and high novelty (Overall solution coverage: 0.351)										
HH1'	$scope * numd * \sim pm * \sim len$	$pa * au$	$pa * au$	Strong	Strong	Moderate	Moderate	Strong	Strong	Moderate
		$pa * au$	$pa * \sim au$	Strong	Strong	Moderate	Moderate	Moderate	**Weak**	**Weak**
HH2'	$scope * numd * pm * \sim len$	$pa * au$	$pa * au$	Strong	Strong	Strong	Strong	Strong	Strong	Strong
		$pa * \sim au$	$pa * au$	Strong	Strong	Moderate	Strong	**Weak**	Strong	Strong

Note: For **contingency**, robust indicates that the collective-attention configuration is a solution configuration or is the best solution(s) in "difficult" situations across all possible contingency combinations; $scope$ indicates a broad scope of innovative work; $numd$ indicates many developers; pm refers to mature project stage; len indicates long release cycle; $\sim$ indicates the absence. For **collective attention**, pa indicates attention partition; au indicates attention augmentation. For **evidence from robustness tests**: strong means that raw consistency ≥ 0.8 and PRI score ≥ 0.7; moderate means that raw consistency ≥ 0.75 and PRI score ≥ 0.65; weak means that raw consistency ≥ 0.75 but $0.65 >$ PRI score ≥ 0.6 and that solutions in the main results are **not** the best-performing ones regarding PRI scores.

5.1.1 Comprehensive Collective-Attention Configuration: Dual Focus with Two-Way Orientation

Consistent with the prior open-source software development literature (Howison and Crowston 2014), we observe that the majority of work is accomplished with one single programmer working on any one task, meaning that individuals' attention is overall differentiated. However, this is not to say that to achieve high speed or high novelty, a collective should completely differentiate its individuals' attention to tasks (characterized by issues or associated modules), i.e., having attention partition but not attention augmentation. Instead, the configurational analyses considering collective-attention constructs indicate that a combination of attention partition and attention augmentation by both issues and modules is sufficient for achieving the singular goal of high speed or high novelty (see configuration HS1 and HN1). Moreover, when this configuration of collective attention is combined with contingencies of a short release cycle with many developers working on a broad scope of innovative work, it is sufficient to achieve the dual goal of high speed and high novelty. Table 5.2 summarizes related propositions.

Linking this collective-attention configuration to the theoretical framework, we theorize its relationship with high speed (or high novelty) in two steps to explain:

- why a *dual focus of collective attention* (attention partition and attention augmentation) facilitates innovation speed (or novelty) in OSS projects and,
- why a *two-way orientation* of collective attention (issue and module) further benefits innovation speed (or novelty).

We also empirically link the two-way dual focus of collective attention back to our data and discuss related observations. Finally, we reason why appropriate contingent conditions are needed to achieve the dual goal (of both high speed and high novelty) and reason their relationships with the dual goal.

Table 5.2 Propositions on the Relationships Between the Comprehensive (Two-Way Dual Focus) Collective-Attention Configuration and the Three Innovation Productivity Goals

Label	Goal	Collective attention configuration	Contingencies	Interpretation
P1	High speed	Two-way Dual Focus	None	• Attention partition facilitates innovation speed by reducing interdependencies among individuals' work. • Attention augmentation complements attention partition by helping individuals to develop a shared understanding that enables them to search for and evaluate ideas more effectively and efficiently.
P2	High novelty	Two-way Dual Focus	None	• The dual focus of collective attention supports both random variation and creative synthesis. • This dual focus outperforms the alternative scenario with differentiated secondary focus and overlapped primary focus, i.e., the absence of both attention partition and attention augmentation.
P3	High speed and high novelty	Two-way Dual Focus	Short release cycle with many available developers working on a broad scope of innovative work	• Collective attention is not a singular solution for innovation productivity, especially as achieving the goals of high speed and high novelty is more challenging than achieving either one. • Having many developers working on a broad scope of innovative work in a short release cycle favors the attainment of both high speed and high novelty; when these contingencies are co-present with two-way dual focus collective attention, high speed and high novelty can be attained.

Dual Focus of Collective Attention

Recall that attention partition (i.e., pa) depicts the extent to which individuals' primary foci differ in a release cycle of an OSS project and that attention augmentation (i.e., au) depicts the extent to which individuals support other's innovative work with a secondary focus of attention in the focal release cycle. The dual collective-attention configuration (i.e., $pa * au$) corresponds to an organizing mode of differentiating individuals' primary foci but overlapping their secondary foci. We first reason the configuration's relationship with innovation speed and then with innovation novelty.

Dual Focus of Collective Attention and Innovation Speed

A dual focus of collective attention, with attention partition and attention augmentation, facilitates innovation speed (see HS1 in Table 5.1). First, attention partition manages the interdependencies among individuals' work through differentiating individuals' primary foci and reduces the coordination cost, and we argue that it is especially important in the open-source software development due to the distinctive contextual aspects. With unbounded innovation space, multiple individuals can easily complicate a simple problem and drift the issue-solving process away from achieving closure. The problem escalates with fluid innovation agency and unsupervised resource allocation because it is also difficult to construct a timeline or a plan, on which these individuals can execute. In our data, we observe that within a focal release cycle, the rate of resolved issues decreases when an issue has two or more individuals' primary foci overlaps at it (see Table 5.3).

Second, after explaining the benefits of attention partition, we further argue that the inclusion of attention augmentation does not hurt. Although attention augmentation promotes collaboration among individuals, it does not intensify the interdependency problem. Specifically, assume that all individuals' primary foci are ideally differentiated (that is, no tasks have two or more individuals regard it as primary focus). If two or more individuals' attention to a task (e.g., an issue) overlaps, it can overlap in only two modes: (i) primary-secondary, i.e., having one regard the task as primary focus while others regard it as their secondary foci and (ii) secondary-secondary, i.e., having all involved individuals regard it as their secondary foci.

In a primary-secondary mode, it is very likely that the one who regards the task as the primary focus plays a core role in accomplishing the task while others support this developer’s work. With such a clear picture of roles, the interdependencies among individuals’ work become quite manageable. In a secondary-secondary mode, the interdependencies among individuals’ work are less likely to slow down the overall innovation process. Every individual has another task as the primary focus. Should frictions (such as waiting time for others’ work) occur during the collaboration, individuals can easily pivot to tasks that are their primary foci without increasing the overall coordination cost. In our data, we observe that the resolved rate of issues in the primary-secondary mode is only slightly lower than the average resolved rate of all issues (i.e., 63% < 67%) and that the resolved rate of issues in the secondary-secondary mode is higher than the average resolved rate (i.e., 73% > 67%), as shown in Table 5.3.

Third, the two collaboration modes, which emerge with attention partition and attention augmentation, accelerate collective innovation by helping individuals develop a shared understanding of elements involved in the software development of a focal release cycle. That understanding can act as a map with which individuals can search for and evaluate ideas in a more effective and efficient way. Take the issue orientation as an example. Building a map on connections among issues helps individuals to develop assumptions and rules on issue-solving approaches. These assumptions and rules can enable individuals to generate ideas for solutions more effectively and more efficiently. As a result, more useful software updates are likely to be created. Besides, the map is also likely to help individuals discern useless issue-solving processes at an early stage and avoid futile innovative work. For instance, in our sample, we observe that a noticeable number of outputs of pull requests were rejected because the underlying issue was already resolved in another software update. With a shared understanding of issues, we expect that this type of problem can be easily addressed.

Table 5.3 Differentiating Individuals' Attention to Issues

Types of attention	Description	# of issues	# of resolved issues	Rate of resolved issues
Primary	One individual regards the focal issue as primary focus.	35,860	19,255	53.69%
Secondary	One individual regards the focal issue as secondary focus.	133,075	94,524	71.03%
Primary-primary	Two or more individuals regard the focal issue as their primary foci.	1,242	504	40.58%
Primary-primary-secondary	Two or more individuals regard the focal issue as their primary foci, whereas others regard it as their secondary foci.	1,880	600	31.91%
Primary-secondary	One individual regards the focal issue as primary focus, whereas others regard it as their secondary foci.	5,317	3,358	63.16%
Secondary-secondary	Two or more individuals regard the focal issue as their secondary foci.	9,137	6,665	72.95%
Total		186,511	124,906	66.97%

Dual Focus of Collective Attention and Innovation Novelty

A dual focus of collective attention improves innovation novelty (see HN1 in Table 5.1) by supporting *both* random variation *and* creative synthesis in the open-source software development process. In collective innovation, two mechanisms are important for generating novelty or creative outputs. They are random variation and creative synthesis. Drawn on an evolutionary model, random variation stresses variety (Staw 2009). Attention partition benefits random variation. Differentiating individuals' primary foci on issues or modules facilitates deep exploitation and broad exploration for revisions of current software, which increases the collective's chance of generating breakthrough software updates. Drawn on a dialectic model, creative synthesis proposes to build on similarity and stresses conflict (Harvey 2014). Attention augmentation benefits creative synthesis. Overlapping individuals' secondary foci on issues or modules facilitates conflicts (or divergent ideas), which increases the opportunity for a novel synthesis to form. Novel syntheses lead to breakthrough outputs, specifically software updates in this study. Overall, a combination of attention partition and attention augmentation (i.e., $pa * au$) improves innovation novelty by facilitating random variation and creative synthesis.

Consider the alternative scenario: overlapping individuals' primary foci and differentiating their secondary foci (i.e., $\sim pa * \sim au$). In this case, overlapping individuals' primary foci (or having the absence of attention partition), rather than secondary foci, may be seen as a quicker way to create conflicts. However, individuals in an open-source collective context are free to enter and leave, and they lack well-established mechanisms to manage conflicts as they can in a formal team context in firms; in formal teams, they can, for example, implement five conflict-handling modes, delineated by Rahim (1983, 2015) and Thomas (1976, 1992). Accordingly, we conjecture that fierce conflicts among individuals, arising from overlapping individuals' primary foci, can lead to frictions and coordination costs and also lead to rapid attrition or disengagement, thereby backfiring on collective innovation.

Similarly, although differentiating individuals' secondary foci (or having the absence of attention augmentation), rather than their primary foci, seems to facilitate a broader exploration of output varieties, the exploration is relatively superficial. We conjecture that the chances of generating useful outputs with a broad scope of superficial variation can be low and that the futile effort can reduce individuals' efficacy on work with their primary foci. These conjectures are supported by the empirical evidence on unfavorable outcomes: the absence of attention partition and attention augmentation by both issues and modules is sufficient to elicit low speed or low novelty (see configuration LS1-2 and LN1 in Table 4.5, Chapter 4)

Two-Way Orientation of Collective Attention

If a collective's attention is oriented toward both issues and modules, it has a two-way orientation. This issue-and-module orientation involves allocating attention with a consideration of the connection between issues and modules. Take an extreme example where in a collective, individuals' primary foci are differentiated by issues, but the issues that are individuals' primary foci are all associated with the same module. While there is attention partition by issues, there is no attention partition by modules. Individuals' work has very high interdependency by modules. The consequent coordination cost can slow down the generation of software updates. In contrast, with a two-way orientation, the effects of a dual focus of collective attention on achieving innovation speed or novelty are assured to play out.

In summary, based on our findings for HS1 and HN1 (i.e., for HS1, $pai * pam * aui * aum$ → high speed; for HN1, $pai * pam * aui * aum$ → high novelty) and the above interpretation, we propose the following propositions on this two-way dual focus of collective attention.

Proposition 1*. A dual focus of collective attention (attention partition and attention augmentation) with a two-way orientation (issues and modules) achieves high speed of innovation productivity in a release cycle of an open-source software project.*

Proposition 2*. A dual focus of collective attention (attention partition and attention augmentation) with a two-way orientation (issues and modules) achieves high novelty of innovation productivity in a release cycle of an open-source software project.*

Two-Way Dual Focus of Collective Attention and the Dual Goal

Although we do not have strong empirical evidence that the two-way dual focus of collective attention is sufficient to achieve the dual goal of both high speed and high novelty, this configuration is the best-performing one (among 16 possible ones) based on both raw consistency and PRI consistency. Specifically, the raw consistency and PRI consistency are 0.69 and 0.56 respectively (relative to the threshold of 0.80 for raw consistency and 0.70 for PRI consistency), while the next best configuration has a raw consistency and PRI consistency of 0.65 and 0.43. This next best configuration has the absence of attention augmentation by issues but the presence of the other three elements (i.e., $pai * pam * {\sim}aui * aum$). We observe that the consistency performance of configurations decreases as they have more elements of the two-way dual collective-attention configuration as absent. Thus, the two-way dual focus of collective attention may be the most promising one for achieving the dual goal. However, the attainment of the dual goal of high speed and high novelty needs additional conditions to be part of the configuration solution.

By taking contingencies into consideration, we get a better understanding on what the additional conditions are. The two-way dual focus of collective attention is sufficient for achieving the dual goal in the context of a short release cycle (i.e., ${\sim}len$) with many developers (i.e., $numd$) working on a broad scope of innovative work (i.e., $scope$) (see HH1 and HH2, where we simplify the contingencies by removing project maturity; keeping other contingencies as the same, project maturity can be either present or absent for the two-way dual collective-attention configuration to achieve the dual goal). The need for appropriate contingencies indicates that collective attention allocation is not a panacea, especially when the goal is hard to attain and the situation is “difficult”. For instance, in a “difficult” situation of having few developers working on a limited scope of innovative work in a release cycle, even if a collective employs the comprehensive configuration (two-way dual focus) of collective attention, its speed of producing outputs and novelty of produced outputs may still not surpass that of a collective employing a slightly different configuration in a “favorable” situation of having many developers working on a broad scope of innovative work in a release cycle.

We consider a short release cycle with many developers working on a broad scope of innovative work (i.e., $scope * numd * {\sim}len$) favorable for several reasons. First, a short release cycle establishes a tight time goal for the release, which is likely to help available developers work in a focused manner on the project. Pacing to meet a short timeline for deliverables is likely to accelerate innovation speed by reducing back-and-forth reworking or postponing. A near-term timeline manner also creates constraints that can generate creative ideas and benefit innovation novelty. Second, the participation of developers involved in a large scope of innovative work makes it feasible for the work to be partitioned and augmented in the short timeframe for the release cycle; otherwise, the possibilities of partitioning and augmenting individuals’ work would be largely constrained. The favorable situation makes it possible for the effect of two-way dual focus of collective attention to be fully played out.

In summary, based on our findings for HH1 and HH2 (i.e., for HH1, $pai * pam * aui * aum * scope * numd * \boldsymbol{{\sim}pm} * {\sim}len$ → both high speed and high novelty; for HH2, $pai * pam * aui * aum * scope * numd * \boldsymbol{pm} * {\sim}len$ → both high speed and high novelty) and the above interpretation, we propose the following proposition on the two-way dual focus of collective attention and the accompanying contingencies to achieve the dual goal of innovation productivity.

Proposition 3*. For a short release cycle of an open-source software project with many developers working on a broad scope of innovative work, a dual focus of collective attention (attention partition and attention augmentation) with two-way orientation (issues and modules) achieves the dual goal of high speed and high novelty for innovation productivity in the release cycle.*

5.1.2 Equifinal Collective Attention Configurations: Redundant Collective Attention Elements and Contingencies

We now interpret our results to decipher equifinal solutions where one or more aspect of collective attention may not be needed to achieve a particular innovation-productivity goal. Although the two-way dual focus emerges an ideal solution for all three productivity goals, some element(s) in the configuration may be redundant for a particular goal or under given contingencies. Like configuration design in engineering, elements in the configuration can play different roles based on contingencies and goals. It is important to discern whether the presence or absence of element(s) changes the attainment of a goal given specific contingencies. Thus, we further develop our theory by interpreting the redundancy of elements in collective-attention configurations under different contingencies to attain a goal and formulating corresponding propositions.

Table 5.4 Propositions on Redundant Collective Attention Elements and Contingencies

Label	Goal	Contingencies	Redundant element(s) in the two-way dual focus of collective attention	Interpretation
P4	High speed	None	Attention augmentation by issues	• Attention augmentation plays a supportive role in attaining high speed, so a one-way orientation of attention augmentation is sufficient. • In the lens of speed, module-oriented attention augmentation is more relevant than issue-oriented attention augmentation.
P5a	High speed	Short release cycle with many available developers working on a broad scope of innovative work	Attention augmentation by issues and by modules	• The benefits of having many developers working on a broad scope of innovative work in a short release cycle for speed compensate the loss of not augmenting attention by issues and modules.
P5b	High speed	Long release cycle at a mature OSS project stage with many developers working on a broad scope of innovative work	Attention augmentation by modules	• Project maturity and attention augmentation can mitigate the risk of a long release cycle. • The benefits of many available developers and a broad scope of innovative work make it possible to achieve speed with one-way attention augmentation; as the project is at a mature stage, attention augmentation by issues is more important.
P5c	High speed	Short release cycle at a mature OSS project stage with many available developers working on a limited scope of innovative work	Attention augmentation by modules	• The benefits of having many developers working on a broad scope of innovative work in a short release cycle for speed compensates the loss of not augmenting attention by issues and modules. • A dual focus of collective attention by issues can migrate the risk of working on a limited scope of innovative work by increasing the resolved rate of issues.
P6a	High novelty	Many available developers working on a broad scope of innovative work	Attention augmentation by issues or by modules	• Diversity elicited by the participation of many developers and the engagement with a broad scope of innovative work compensates the loss of not overlapping individuals' secondary foci by either issues or modules.
P6c	High novelty	Long release cycle with many available developers working on a broad scope of innovative work	Attention partition by modules	• A long release cycle accommodates the coordination cost caused by the absence of attention partition by modules but also makes synthesizing by two-way augmentation more important.

Note: Although we find equifinal collective-attention solutions for the dual goal in our main analysis, which indicate the redundancy of attention augmentation either by issues or by modules (see HH1' and HH2' in Table 5.1), those solutions are not well supported by a range of robustness tests. In some robustness tests, the evidence is week. Therefore, we do not theorize on the redundancy of those elements for achieving the dual goal.

Equifinal Solutions for High Speed as the Innovation Productivity Goal

For achieving high speed, configuration HS1 indicates that the presence or absence of attention augmentation by issues (i.e., aui) does not make a difference when the other three elements (i.e., $pai * pam * aum$) in the two-way dual focus of collective attention are present. Without attention augmentation by issues, the other three elements are still sufficient for achieving high speed. Thus, we infer that managing interdependencies among individuals' work by attention partition (i.e., differentiating their primary foci) plays a critical role for achieving high speed overall. Attention augmentation complements attention partition with two effective modes (i.e., primary-secondary and secondary-secondary modes) for individuals to collaborate in tasks, but it plays a supportive role. Thus, attention augmentation can be either implemented in a one-way orientation (by issues or by modules) or not implemented at all. However, among these implementation ways, we find that only one-way orientation by modules demonstrates a contingency-robust solution for high speed when combined with attention partition by both issues and files.

Augmentation becomes important when one individual does not have sufficient knowledge to accomplish the task. It can be the lack of knowledge on related modules within the codebase or the lack of ideas on how to solve the problem underlying the task. If the goal is just to finish the task with one solution rather than to finish the task with the best solution, supplementing the necessary knowledge on related modules is likely more relevant or pragmatic than generating more ideas on solutions for the problem. Module-oriented attention augmentation connects to the former, whereas issue-oriented attention augmentation connects to the latter. As such, if the lens is just on speed (i.e., finishing the tasks as soon), attention augmentation by issues emerges as a superfluous condition (presence or absence does not hurt or help).

In summary, based on our findings for HS1 (i.e., $pai * pam * aum$ → high speed) and the above interpretation, we propose the following proposition on the redundancy of attention augmentation by issues (i.e., aui).

***Proposition 4**. For achieving high speed of innovation productivity in a release cycle of an open-source software project, there is no difference in the presence or absence of attention augmentation by issues when attention partition by issues and by modules as well as attention augmentation by modules are present.*

Configurations HS1'-HS3' further indicate that attention augmentation by modules (i.e., aum) or both attention augmentation by issues and attention augmentation by modules (i.e., aui and aum) can be redundant under some contingencies.

Configuration HS1' indicates that for a focal release cycle which is short (i.e., $\sim len$) and has many developers (i.e., $numd$) working on a broad scope of innovative work (i.e., $scope$), both attention augmentation by issues and attention augmentation by modules is redundant if attention partition by issues and by modules is present. In other words, in the situation with given contingencies, a collective does not need to overlap individuals' secondary foci to gain benefits from augmentation, although managing interdependencies by differentiating individuals' primary foci is still important. The effect of the contingencies can compensate the loss of not having attention augmentation. We argue that it is feasible for a lens on only speed. Among all eight possible combinations of the three contingencies (scope of innovative work, availability of developers, and the length of release cycle) in terms of presence and absence, the combination of a short release cycle, many developers, and a broad scope of innovative work (i.e., $scope * numd * {\sim}len$) is the most favorable situation for innovation speed (as reasoned in proposition P3). Thus, it is highly probable that a collective can achieve high speed without attention augmentation in this most favorable situation.

In summary, based on our findings for HS1' (i.e., $pai * pam * scope * numd * {\sim}len$ → high speed) and the above interpretation, we propose the following proposition on the redundancy of attention augmentation by issues (i.e., aui) and attention augmentation by modules (i.e., aum).

***Proposition 5a**. For achieving high speed of innovation productivity in a short release cycle for an open-source software project with many developers working on a broad scope of innovative work, there is no difference in the presence or absence of attention augmentation when attention partition by issues and modules is present.*

Configuration HS2' illuminates the role of attention partition and attention augmentation in a new situation, i.e., a long release cycle (i.e., len) at a mature project stage (i.e., pm) with many developers working on a broad scope of innovative work. In this situation, there is no difference in the presence or absence of attention augmentation by modules when attention partition by issues and by modules as well as attention augmentation by issues are present. Among the contingent conditions, the long release cycle may be a glaring element because the collective's efficacy can decrease with a distant milestone. However, project maturity and attention augmentation can mitigate this risk. A project at a mature stage is expected to have an established codebase architecture. With this established architecture and attention augmentation, it is possible that a change in one part can then lead to changes in the second part, and these changes can result in further changes in other parts, and so on. The resultant "adaptive walk" is likely to sustain the collective's innovation efficacy.

Besides, the other favorable contingencies, i.e., many available developers working on a broad scope of innovative work, make it possible to achieve high speed with only one-way (instead of two-way) orientation of attention augmentation. Compared with attention augmentation by issues, attention augmentation by modules is no longer important as developers are likely to know the codebase well when the project is at a mature stage.

In summary, based on our findings for HS2' (i.e., $pai * pam * aui * scope * numd * pm * len$ → high speed) and the above interpretation, we propose the following proposition on the redundancy of attention augmentation by modules (i.e., aum).

Proposition 5b*. For achieving high speed of innovation productivity in a release cycle that is long and is for a mature open-source software project with many developers working on a broad scope of innovative work, there is no difference in the presence or absence of attention augmentation by modules when attention partition by issues, attention partition by modules, and attention augmentation by issues are present.*

Configuration HS3' reveals that for a short release cycle at a mature project stage with many developers working on a limited scope of innovative work (i.e., ~scope), attention augmentation by modules is redundant. Among the contingent conditions, a limited scope of innovative work is relatively glaring. However, as we mentioned above, a dual focus of collective attention is likely to increase the resolved rate of issues. If the resolved rate of issues is increased, the limited scope of innovative work may not be a problem. Besides, similarly to the reasoning for configuration HS2', attention augmentation by modules can be no longer important given that the project is at mature stage and that the focal release cycle is short and has many developers working on it.

In summary, based on our findings for HS3’ (i.e., $pai * pam * aui * {\sim}scope * numd * pm * {\sim}len$ → high speed) and the above interpretation, we propose the following proposition on the redundancy of attention augmentation by modules (i.e., aum).

Proposition 5c. *For achieving high speed of innovation productivity in a release cycle that is short and is for a mature open-source software project with many developers working on a limited scope of innovative work, there is no difference in the presence or absence of attention augmentation by modules when attention partition by issues, attention partition by modules, and attention augmentation by issues are present.*

Equifinal Solutions for High Novelty as the Innovation Productivity Goal

For innovation novelty, configurations HN1’-HN3’ indicate that under certain contingencies, the presence or absence of one element on attention partition or attention augmentation does not make a difference when the other three elements in the two-way dual focus of collective attention are still present.

Configurations HN1’ and HN2’, which have the same contingencies, differ only with respect to attention augmentation. In the case of HN1’, attention augmentation by modules is redundant, whereas in the case of HN2’, attention augmentation by issues is redundant. Taken together, they indicate that for achieving high novelty in a release cycle with many developers working on a broad scope of innovative work, either attention augmentation by issues or attention augmentation by modules is redundant when attention partition by issues and by modules is present. Recall that the comprehensive collective attention configuration is sufficient for achieving high novelty by facilitating both random variation and creative synthesis in a collective. Configurations HN1’ and HN2’ indicate that the two mechanisms are still needed, but attention augmentation can be implemented in a one-way orientation. The reason is that the situation of a release cycle with many developers working on a broad scope of innovative work is a situation that favors the attainment of high novelty; the active experiments on various potential software updates lead to a high probability that a breakthrough idea comes out. In the lens of only high novelty, advantages of the novelty-favorable situation can compensate the loss of not overlapping individuals’ secondary foci by either issues or modules.

In summary, based on our findings for HN1’ and HN2’ (i.e., for HN1’, $pai * pam * \boldsymbol{aui} * scope * numd$ → high novelty; for HN2’, $pai * pam * \boldsymbol{aum} * scope * numd$ → high novelty) and the above interpretation, we propose the following proposition on the redundancy of either attention augmentation by modules (i.e., aum) or attention augmentation by issues (i.e., aui).

***Proposition 6a**. For achieving high novelty of innovation productivity in a release cycle of an open-source software project with many developers working on a broad scope of innovative work, there is no difference in the presence or absence of attention augmentation by issues (or by modules) when attention partition by issues, attention partition by modules, and attention augmentation by modules (or by issues) is present.*

Configuration HN3’, which includes long release cycle as an additional contingency to those in HN1’ and HN2’, suggests that there is no difference in novelty with the presence or absence of attention partition by modules as part of the collective attention. When the release cycle is long and only novelty (of software updates) is under consideration, individuals have plenty time to accommodate the coordination cost. Even if the issues to which individuals allocate their attention with a primary focus are associated with few modules (meaning that individuals' primary foci are overlapped by modules), the collective can still achieve high novelty as long as its individuals differentiate their primary foci on issues and overlap their secondary foci on issues and modules with others' attention to facilitate both random variation and creative synthesis. However, without a tight time goal for the release, developers' work for the OSS project is likely to be fragmented. In this situation, synthesizing with a two-way attention augmentation becomes important. This may be the reason why we do not observe the redundancy of attention augmentation by issues or by modules even though the release cycle has many developers working on a broad scope of innovative work just as we do in configuration HN1' and HN2'.

In summary, based on our findings for HS3' (i.e., $pai * aui * aum * scope * numd * len$ → high novelty) and the above interpretation*,* we propose the following proposition on the redundancy of attention partition by modules (i.e., pam).

Proposition 6b*. For achieving high novelty of innovation productivity in a release cycle that is long and is for an open-source software project with many developers working on a broad scope of innovative work, there is no difference in the presence or absence of attention partition by modules when attention partition by issues, attention augmentation by issues, and attention augmentation by modules are present.*

5.2 THEORETICAL IMPLICATIONS

The collective attention view proposed in this study is useful and valid to conceptualize collective-level constructs that characterize open-source software development. By doing so, we add to prior literature that has specified individual-level self-organizing mechanisms for open-source software development (e.g., Lindberg et al. 2016; Howison and Crowston 2014) but understudied the development of collective-level constructs.

First, it takes a holistic but granular approach (i.e., it observes granular actions conducted by all individuals in a collective) to access the essence of a collective's selection (or collective attention) and thus, accommodates distinctive contextual aspects of collective innovation in the open-source context. Leveraging the comprehensive observability of individuals' actions in OSS projects, the collective attention view conceptualizes distribution pattern of those granular actions in a behavior space. The distribution pattern, as realized mapping route between a collective's available development resources and its innovation space, speaks to regularities hidden in individuals' interactive actions. It makes no assumptions on available resources, innovation space, and the existence of managerial force guiding the mapping between the available resources and the innovation space.

By focusing on granular innovative actions of individuals in a collective, we define a behavior space with three dimensions (i.e., developers, issues, and modules) to depict the collective's distribution of innovative actions. Each innovative action indicates an individual's selection about where to allocate their available resources (i.e., to which issues or modules) at a certain time point. Thus, a collection of all individuals' innovative actions in a period (specifically, a release cycle) constructs a holistic picture on the mapping route between the collective's available resources to potential innovative work, i.e., how the collective organizes their innovative work.

We approach the hidden regularity on the interdependency between individuals' selection (i.e., how a collective differentiates and overlaps its individuals' attention) by characterizing the distribution pattern of their innovative actions.

Second, the collective attention view enables researchers to develop meaningful parsimonious constructs, which conceptualize the spontaneous order emerging from individual actions and interactions, relying on platform-based functionalities and associated practices. In our study, we conceptualize four constructs to contrast collectives' organizing modes. We differentiate between an individual's primary focus and secondary focus of attention based on the distribution of the individual's innovative actions. Attention partition (i.e., pa) depicts the extent to which individuals' primary foci differ in a given release cycle of an OSS project, whereas attention augmentation (i.e., au) depicts the extent to which individuals support others' work with their secondary foci in the release cycle. After further differentiating whether a collective allocates its attention by issues (producing pai and aui) or by modules (producing pam and aum), we conceptualize four constructs to characterize 16 possible hidden organizing modes (according to the presence and absence of each characteristic), depicting whether a collective differentiates or overlaps individuals' primary and secondary foci by issues or by modules.

Finally, we discern how the collective-attention constructs explain different innovation outcomes in open-source software development. With these constructs, we obtain insights on how collectives that achieve different innovation productivity goals (i.e., high speed, high novelty, and both high speed and high novelty) are organized differently from the others. The explanatory power of the solution configurations with these collective-attention constructs is reasonably high. The coverage of the contingency-robust solution of collective attention for achieving high speed and for achieving high novelty (i.e., HS1 and HN1) is 0.46 and 0.35 respectively, meaning that about 46% high-speed outcome can be explained by HS1 and that about 35% high-novelty outcome can be explained by HN1. The overall coverage of solution configurations with collective attention and contingencies under consideration for each innovation productivity goal (i.e., high speed, high novelty, or both) is also reasonably high, ranging from 0.35 to 0.44.

Besides, the four constructs effectively decipher how collectives that achieve the three productivity goals are organized differently, especially those that achieve the dual goal. For a singular goal of high speed or high novelty, a two-way dual focus of collective attention (i.e., attention partition and attention augmentation by issues and by modules) is sufficient, and this solution is contingency robust (as shown in HS1 and HN1). We also find that with favorable contingencies (such as a short release cycle with many developers working on a broad scope of innovative work), the sibling configurations with one element (or two elements) as absent can also elicit the singular goal, indicating the redundancy of the element(s). For the dual goal, we do not find contingency-robust solution of collective attention. However, the analyses considering only collective-attention constructs reveal that 13 of 16 possible collective-attention configurations do not achieve the dual goal. In other words, they elicit low speed or low novelty. Taking contingencies into consideration, we decipher contingencies under which the dual goal can be achieved through the two-way dual focus of collective attention. As such, this requires a release cycle to be short and have many developers working on a broad scope of innovative work (involving many issues and many modules).

5.3 PRACTICAL IMPLICATIONS

Our study offers three practical implications which pertain to metrics that can be incorporated in open-source platforms, guidelines on organizing individuals work toward achieving collective productivity goals in OSS projects, and contingencies that favor the achievement of high speed or high novelty in open-source software development.

First, open-source platform owners like GitHub can use the collective-attention metrics to build greater awareness and shared understanding among individuals of a collective on how they are organizing their work relative to their goals. While these platforms are committed to offering metrics that help developers to better understand how to improve their productivity in OSS projects, most existing metrics are too detailed, i.e., tied to a specific aspect of open-source software development. For instance, GitHub Insights includes metrics on pull-request size, which enables developers to detect and remove large pull requests that are very risky to deploy and difficult to review, merge, and release. These metrics are direct, and improvements that they imply are actionable. However, they cannot build awareness on how a collective is organizing its work overall and awareness on the relation between its organizing modes with productivity goals, regarding the speed of producing software updates and the novelty of produced updates. The four collective attention constructs we develop in this study fill this gap. For specific productivity goals, the configurational solutions, discovered by the four constructs, are also interpretable and powerful. They can function as reference organizing modes for innovation productivity goals.

Second, for OSS project stakeholders, we contribute useful insights on how to organize individuals' work toward the attainment of collective innovation-productivity goals regarding speed and novelty. Specifically, we find that the ideal organizing mode for collectives to achieve innovation productivity in OSS projects is differentiating individuals' primary foci but overlapping individuals' secondary foci by both issues and modules (i.e., $pai * pam * aui * aum$). However, we observe that the collective-attention configuration with the second largest number of cases (about 18% cases) is a mirror image of the ideal organizing mode, i.e., $\sim pai * \sim pam * \sim aui *$

$\sim aum$. Collectives in these cases overlap individuals' primary foci and differentiate individuals' secondary foci by both issues and modules. The analytical results reveal that this organizing mode elicits low speed (or low novelty) with high consistency. Our study offers guidelines for such collectives on how to get out of the productivity predicament, that is to differentiate individuals' primary foci and overlap their secondary foci by both issues and modules.

Finally, we also offer insights on contingencies that favor the attainment of high speed or high novelty in OSS projects, namely, situations under which some required elements of collective attention (or ideal organizing modes) can be out of concern. Specifically, for a singular goal on speed, if a focal release cycle is short and has many developers working on a broad scope of innovative work (i.e., $scope * numd * \sim len$), the collective just needs to differentiate its individuals' primary foci by issues and modules. In other words, keeping attention partition by issues and modules, it does not differ whether they overlap individuals' secondary foci, meaning that attention augmentation can be completely out of concern. For a singular goal on novelty, if a focal release cycle has many developers working on a broad scope of innovative work (i.e., $scope * numd$), the collective still needs to partition its individuals' primary foci by both issues and modules. However, as to attention augmentation, the collective needs to overlap its individuals' secondary foci either by issues or by modules, meaning that a one-way attention augmentation is sufficient.

5.4 LIMITATIONS AND FUTURE RESEARCH

Our research takes an important first step toward developing a collective attention view on the collective-level determinants for achieving innovation productivity in open-source software development. Consistent with our research objectives, we use a configurational approach to accommodate the complex causality, in which the combinational (rather than net) effects of causes are theoretically meaningful and the combinations (or “recipes”) of causes that elicit a specific outcome manifest equifinality as well as causal asymmetry.

We acknowledge some limitations of the work. First, we discover collective-attention configurations that elicit favorable productivity outcomes but do not offer insights on the formation of these configurations. We encourage future research to investigate antecedents of collective attention allocation, for example, platform mechanisms that can facilitate attention partition and attention augmentation.

Second, we foreground collective attention allocation in execution (where actions of interest pertain to commits that change the codebase), and background collective attention allocation in ideation (where actions pertain to issue detection, comments, and code reviews). Future research can foreground collective attention allocation in ideation and assess its impact on innovation productivity. Additionally, future research can investigate how collective attention allocation in execution can be employed in combination with collective attention allocation in ideation to achieve innovation productivity goals.

Third, our unit of analysis is a release cycle of an OSS project. Future research can examine the dynamics of collective attention across release cycles of an OSS project. As our analytical results do indicate that effective configurations of collective attention vary with temporal aspects such as project maturity, it would be interesting to investigate how the temporal sequence in which an OSS project employs different collective-attention configurations impacts project success, e.g., the market value of the delivered software or the continued engagement of developers.

Fourth, we use lines of code change to proxy attention allocation. The more lines of code change an individual contributes for solving an issue (modifying a module), the more attentive the individual is to the issue (the module). This measure is reasonable but not precise. Work time can be a more precise proxy, but we are unable to observe it in our empirical setting. Future research can apply the collective attention view in an empirical setting or experimental setting, where individuals' work time to an issue or a module can be measured.

Fifth, we do not differentiate the nature of issues (for example, in levels of difficulty) to be addressed in a release cycle of an OSS project. In the future research, it would be interesting to investigate how the effective configurations of collective attention for different innovation productivity goals vary with contingencies related to the nature of issues.

Finally, our empirical setting is GitHub OSS projects on the machine-learning topic. There may be contextual aspects at the level of the platform or topic of the OSS projects that restrict the generalization of our findings. We encourage future research to evaluate how contextual aspects at the level of the open-source platform and the topic of the projects will affect the relationship between collective attention allocation and innovation productivity goals.

REFERENCES

Ågerfalk, P. J., and Fitzgerald, B. 2008. “Outsourcing to an Unknown Workforce: Exploring Opensourcing as a Global Sourcing Strategy,” *MIS Quarterly* (32:2), pp.385-409.

Aguinis, H., and O’Boyle Jr., E. 2014. “Star Performers in Twenty-First Century Organizations,” *Personnel Psychology* (67:3), pp. 313-350.

Ahuja, G., Lampert, C.M., and Tandon, V. 2008. “Moving Beyond Schumpeter: Management Research on the Determinants of Technological Innovation,” *Academy of Management Annals* (*2*:1), pp.1-98.

Ancona, D. G., and Caldwell, D.F. 1992. “Demography and Design: Predictors of New Product Team Performance,” *Organization Science* (*3*:3), pp.321-341.

Bagozzi, R. P., and Dholakia, U.M. 2006. “Open Source Software User Communities: A Study of Participation in Linux User Groups,” *Management Science* (52:7), pp.1099-1115.

Baldwin, C. Y., and Clark, K.B. 2000. *Design Rules: The Power of Modularity* (Vol. 1), Massachusetts, Cambridge: MIT Press.

Bansal, P., Kim, A., and Wood, M. O. 2018. “Hidden in Plain Sight: The Importance of Scale in Organizations’ Attention to Issues,” *Academy of Management Review* (43:2), pp. 217-241.

Barnett, M. L. 2008. “An Attention-Based View of Real Options Reasoning,” *Academy of Management Review* (33:3), pp. 605–628.

Benner, M. J., and Tushman, M.L., 2015. “Reflections on the 2013 Decade Award—'Exploitation, Exploration, and Process Management: The Productivity Dilemma Revisited’ Ten Years Later,” *Academy of Management Review* (40:4), pp. 497-514.

Blau, P. M. 1977. *Inequality and Heterogeneity*, New York: Free Press.

Campbell, D. T. 1960. “Blind Variation and Selective Retentions in Creative Thought as in Other Knowledge Processes,” *Psychological Review* (*67*:6), p.380-400.

Campbell, J. T., Sirmon, D.G., and Schijven, M. 2016. “Fuzzy Logic and the Market: A Configurational Approach to Investor Perceptions of Acquisition Announcements,” *Academy of Management Journal* (59:1), pp.163-187.

Cronin, M. A., & Weingart, L. R. 2007. “Representational Gaps, Information Processing, and Conflict in Functionally Diverse Teams,” *Academy of Management Review* (32:3) 761–773.

Dahlander, L., and O'Mahony, S. 2011. “Progressing to the Center: Coordinating Project Work,” *Organization Science* (22:4), pp.961-979.

Dong, J. Q., Wu, W., and Zhang, Y. S. 2019. “The Faster the Better? Innovation Speed and User Interest in Open Source Software,” *Information & Management* (56:5), pp. 669-680.

Doty, D. H., Glick, W. H., and Huber, G. P. 1993. “Fit, Equifinality, and Organizational Effectiveness: A Test of Two Configurational Theories,” *Academy of Management Journal* (36:6), pp. 1196-1250.

Drazin, R. and Van de Ven, A. H. 1985. “Alternative Forms of Fit in Contingent Theory,” *Administrative Science Quarterly* (30:4), pp. 514-539,

El Sawy, O. A., Malhotra, A., Park, Y., and Pavlou, P.A. 2010. “Research Commentary—Seeking the Configurations of Digital Ecodynamics: It Takes Three to Tango,” *Information Systems Research* (21:4), pp.835-848.

Ethiraj, S. K., and Levinthal, D. 2004. “Modularity and Innovation in Complex Systems,” *Management Science* (50:2), pp. 159-173.

Fiss, P. C. 2007. “A Set-Theoretic Approach to Organizational Configurations,” *Academy of Management Review* (32:4), pp.1180-1198.

Fiss, P. C. 2011. “Building Better Causal Theories: A Fuzzy Set Approach to Typologies in Organization Research,” *Academy of Management Journal* (54:2), pp.393-420.

George, A. 1979. “Case Studies and Theory Development: The Method of Structured, Focused Comparison,” in *Diplomacy: New Approaches in History, Theory and Policy*, P. G. Lauren (eds.), New York: Free Press.

George, A., and Bennett, A. 2005. *Case Studies and Theory Development*, Cambridge, MA: MIT Press.

Greckhamer, T. “CEO compensation in relation to worker compensation across countries: The configurational impact of country-level institutions,” *Strategic Management Journal* (37:4), pp. 793-815.

Haas, M.R., Criscuolo, P., and George, G. 2015. “Which Problems to Solve? Online Knowledge Sharing and Attention Allocation in Organizations,” *Academy of Management Journal* (58:3), pp.680-711.

Haddad, I., and Warner, B. 2011. “Understanding the Open Source Development Model,” a white paper by *The Linux Foundation*, November. (https://www.ibrahimatlinux.com/uploads/6/3/9/7/6397792/00.pdf, accessed on May 5, 2020).

Hallowell, E.M. 2005. “Overloaded Circuits: Why Smart People Underperform,” *Harvard Business Review*, January, pp. 54-62.

Harrison, D.A., and Klein, K.J. 2007. “What's the Difference? Diversity Constructs as Separation, Variety, or Disparity in Organizations,” *Academy of Management Review* (32:4), pp.1199-1228.

Harvey, S. 2014. “Creative Synthesis: Exploring the Process of Extraordinary Group Creativity,” *Academy of Management Review* (39:3), pp.324-343.

Heracleous, L., and Barrett, M. 2001. “Organizational change as discourse: Communicative actions and deep structures in the context of information technology implementation,” *Academy of Management Journal* (44:4), pp. 755–778.

Ho, S.Y., and Rai, A. 2017. “Continued Voluntary Participation Intention in Firm-Participating Open Source Software Projects,” *Information Systems Research* (28:3), pp.603-625.

Horwitz, S.K., and Horwitz, I.B., 2007. “The Effects of Team Diversity on Team Outcomes: A Meta-Analytic Review of Team Demography,” *Journal of Management* (33:6), pp.987-1015.

Howison, J., and Crowston, K. 2014. “Collaboration through Open Superposition: A Theory of the Open Source Way,” *MIS Quarterly* (38:1), pp.29-50.

James, W. 1890. *Principles of Psychology (Vol. 1)*, New York, NY: Henry Holt and Company.

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D.M., and Damian, D. 2016. “An In-Depth Study of the Promises and Perils of Mining GitHub,” *Empirical Software Engineering* (21:5), pp.2035-2071.

Katila, R., and Ahuja, G. 2002. “Something Old, Something New: A Longitudinal Study of Search Behavior and New Product Introduction,” *Academy of Management Journal* (45:6), pp.1183-1194.

Kessler, E. H., and Chakrabarti, A. K. 1996. “Innovation Speed: A Conceptual Model of Context, Antecedents, and Outcomes,” *Academy of Management Review* (21:4), pp. 1143-1191.

Kocienda, K. 2018. *Creative Selection: Inside Apple's Design Process During the Golden Age of Steve Jobs*, New York, NY: St. Martin’s Press.

Kuk, G. 2006. “Strategic Interaction and Knowledge Sharing in the KDE Developer Mailing list,” *Management Science* (52:7), pp.1031-1042.

Lee, G.K., and Cole, R.E. 2003. “From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development,” *Organization Science* (14:6), pp.633-649.

Lehmann, J., Gonçalves, B., Ramasco, J.J., and Cattuto, C. 2012. “Dynamical Classes of Collective Attention in Twitter,” in *Proceedings of the 21st International Conference on World Wide Web*, April, pp. 251-260.

Lavie, N. 2005. “Distracted and Confused?: Selective Attention Under Load,” *Trends in Cognitive Sciences* (9:2), pp.75-82.

Li, Q., Maggitti, P.G., Smith, K.G., Tesluk, P.E., and Katila, R., 2013. “Top Management Attention to Innovation: The Role of Search Selection and Intensity in New Product Introductions,” *Academy of Management Journal* (56:3), pp.893-916.

Lindberg, A., Berente, N., Gaskin, J., and Lyytinen, K. 2016. “Coordinating Interdependencies in Online Communities: A Study of an Open Source Software Project,” *Information Systems Research* (27:4), pp.751-772.

Lingo, E. L., and O'Mahony, S. 2010. "Nexus Work: Brokerage on Creative Projects," *Administrative Science Quarterly* (55:1), pp. 47-81.

Ma, M, and Agarwal, R. 2007. "Through a Glass Darkly: Information Technology Design, Identity Verification, and Knowledge Contribution in Online Communities," *Information Systems Research* (18:1), pp. 42-67.

Mackie, J. L. 1965. "Causes and Conditionals," *American Philosophical Quarterly* (2:4), pp. 245–65.

Maruping L.M., Daniel S.L., Cataldo M. 2019. "Developer Centrality and the Impact of Value Congruence and Incongruence on Commitment and Code Contribution Activity in Open Source Software Communities," *MIS Quarterly* (43:3), pp.951-76.

Medappa, P. K., and Srivastava, S. C. 2019. "Does Superposition Influence the Success of FLOSS Projects? An Examination of Open-Source Software Development by Organizations and Individuals," *Information Systems Research* (30:3), pp. 764-786.

Mocanu, D., Rossi, L., Zhang, Q., Karsai, M., and Quattrociocchi, W. 2015. "Collective Attention in the Age of (Mis) information," *Computers in Human Behavior* (51: Part B), pp.1198-1204.

Mole, C. 2011. *Attention is Cognitive Unison: An Essay in Philosophical Psychology*, New York, Oxford: Oxford University Press.

Nambisan, S., Lyytinen, K., Majchrzak, A., and Song, M. 2017. "Digital Innovation Management: Reinventing Innovation Management Research in a Digital World," *MIS Quarterly* (41:1), pp. 223-238.

Neumann, O. 1987. "Beyond Capacity: A Functional View of Attention," in *Perspectives on Perception and Action*, H. Heuer and A.F. Sanders (eds.), Hillsdale: Lawrence Erlbaum Associates, pp. 361–94.

Obstfeld, D. 2005. "Social Networks, the Tertius Iungens Orientation, and Involvement in Innovation," *Administrative Science Quarterly* (50:1), pp. 100–130.

Ocasio, W. 1997. "Towards an Attention-Based View of the Firm," *Strategic Management Journal* (18:S1), pp.187-206.

Ocasio, W. 2011. "Attention to Attention," *Organization Science* (22:5), pp.1286-1296.

Oh, W., and Jeon, S. 2007. "Membership Herding and Network Stability in the Open Source Community: The Ising Perspective," *Management Science* (53:7), pp.1086-1101.

Park, Y., Fiss, P. C., and El Sawy, O. A. 2020. "Theorizing the Multiplicity of Digital Phenomena: The Ecology of Configurations, Causal Recipes, and Guidelines for Applying QCA," *MIS Quarterly* (44:4), pp. 1493-1520.

Park, Y., and Mithas, S. 2020. "Organized Complexity of Digital Business Strategy: A Configurational Perspective," *MIS Quarterly* (44:1), pp.85-127.

Pashler, H., Johnston, J.C., and Ruthruff, E. 2001. “Attention and Performance,” *Annual Review of Psychology* (52:1), pp.629-651.

Potter, R. E., and Balthazard, P. 2004. “The Role of Individual Memory and Attention Processes during Electronic Brainstorming,” *MIS Quarterly* (28:4), pp. 621-643.

Ragin, C. C. 1987. *The Comparative Method: Moving beyond Qualitative and Quantitative Strategies*, Berkeley: University of California Press.

Ragin, C. C. 2000. *Fuzzy-Set Social Science*, Illinois, Chicago: University of Chicago Press.

Ragin, C. C. 2003. “Recent Advances in Fuzzy-Set Methods and Their Application to Policy Questions,” COMPASSS working paper WP2003-9. (http://www.compasss.org/wpseries/Ragin2003a.pdf, accessed on May 17, 2021).

Ragin, C.C. 2006. “Set Relations in Social Research: Evaluating Their Consistency and Coverage,” *Political analysis* (14:3), pp.291-310.

Ragin, C. C. 2008. *Redesigning Social Inquiry: Fuzzy Sets and Beyond*, Illinois, Chicago: University of Chicago Press.

Ragin, C. C., and Davey, S. 2016. *Fuzzy-Set/Qualitative Comparative Analysis 3.0*. Irvine, California: Department of Sociology, University of California.

Ragin, Charles C. 2018. *User's Guide to Fuzzy-Set/Qualitative Comparative Analysis 3.0*, Irvine, California: Department of Sociology, University of California.

Rahim, M. A. 1983. “A Measure of Styles of Handling Interpersonal Conflict,” *Academy of Management Journal* (26:2), pp. 368-376.

Rahim, M. A. 2015. *Managing Conflict in Organizations (4th edition)*, Routledge.

Ransbotham, S., and Kane, G. 2011. “Membership Turnover and Collaboration Success in Online Communities: Explaining Rises and Falls from Grace in Wikipedia,” *MIS Quarterly* (35:3), pp. 613–627.

Raveendran, M., Puranam, P., and Warglien, M. 2016. “Object Salience in the Division of Labor: Experimental Evidence,” *Management Science* (62:7), pp. 2110-2128.

Ren, Y., Chen, J., and Riedl, J. 2016. “The Impact and Evolution of Group Diversity in Online Open Collaboration,” *Management Science* (62:6), pp.1668-1686.

Roberts, J.A., Hann, I.H., and Slaughter, S.A. 2006. “Understanding the Motivations, Participation, and Performance of Open Source Software Developers: A Longitudinal Study of the Apache Projects,” *Management Science* (52:7), pp.984-999.

Sanchez, R., and Mahoney, J. T. 1996. “Modularity, flexibility, and knowledge management in product and organization design,” *Strategic Management Journal* (17:Winter), pp. 63–76.

Sasahara, K., Hirata, Y., Toyoda, M., Kitsuregawa, M., and Aihara, K. 2013. "Quantifying Collective Attention from Tweet Stream," *PLoS ONE* (8:4), e61823. (https://doi.org/10.1371/journal.pone.0061823, accessed on November 26, 2019).

Schneider, C. Q., and Wagemann, C. 2012. *Set-Theoretic Methods for the Social Sciences: A Guide to Qualitative Comparative Analysis*, New York, NY: Cambridge University Press.

Schubert, T., and Tavassoli, S. 2020. "Product Innovation and Educational Diversity in Top and Middle Management Teams," *Academy of Management Journal* (63:1), pp. 272-294.

Setia, P., Rajagopalan, B., Sambamurthy, V., and Calantone, R. 2012. "How Peripheral Developers Contribute to Open-Source Software Development," *Information Systems Research* (23:1), pp.144-163.

Shah, S. K. 2006. "Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development," *Management Science* (52:7), pp.1000-1014.

Simon, H. A. 1997. *Administrative Behavior: A Study of Decision-Making Processes in Administrative Organizations (4th Edition)*, The Free Press.

Simonton, D. K. 2013. "Creative Thought as Blind Variation and Selective Retention: Why Creativity is Inversely Related to Sightedness," *Journal of Theoretical and Philosophical Psychology* (33:4), pp.253-266.

Staw, B. M. 2009. "Is Group Creativity Really an Oxymoron? Some Thoughts on Bridging the Cohesion-Creativity Divide," in *Research on Managing Groups and Teams. Volume 12: Creativity in Groups*, E. A. Mannix, J. A. Goncalo, and M. A. Neale (eds.), Bingley, UK: Emerald, pp. 311–323.

Sternberg, R. J., and Sternberg, K. 2012. "Chapter 4 – Attention and Consciousness," in *Cognitive Psychology*, California, Belmont: Wadsworth, Cengage Learning.

Thomas, K. W. 1976. "Conflict and Conflict Management," in *Handbook of industrial and Organizational Psychology*, M. D., Dunnette (ed.), Chicago: Rand-McNally, pp. 889-935.

Thomas, K. W. 1992. "Conflict and Conflict Management: Reflections and Update," *Journal of Organizational Behavior* (13:3), pp. 265-274.

Von Krogh, G., Haefliger, S., Spaeth, S., and Wallin, M.W. 2012. "Carrots and Rainbows: Motivation and Social Practice in Open Source Software Development," *MIS Quarterly* (36:2), pp.649-676.

Vuori, T. O., and Huy, Q.N. 2016. "Distributed Attention and Shared Emotions in the Innovation Process: How Nokia Lost the Smartphone Battle," *Administrative Science Quarterly* (61:1), pp.9-51.

Weick, K. E., and Sutcliffe, K.M. 2006. "Mindfulness and the Quality of Organizational Attention," *Organization Science* (17:4), pp.514-524.

Wertheimer, M., and Riezler, K. 1944. “Gestalt Theory,” *Social Research* (11:1), pp. 78-99.

Wu, F., and Huberman, B.A. 2007. “Novelty and Collective Attention,” *Proceedings of the National Academy of Sciences* (104:45), pp.17599-17601.

Wu, W. 2011. “Attention as Selection for Action,” in *Attention: Philosophical and Psychological Essays*, C. Mole, D. Smithies and W. Wu (eds.), New York, Oxford: Oxford University Press, pp 97-116.

Wu, W., 2014. *Attention: New Problems of Philosophy Series*, London and New York: Routledge.

Yayavaram, S., and Ahuja, G. 2008. “Decomposability in Knowledge Structures and Its Impact on the Usefulness of Inventions and Knowledge-Base Malleability,” *Administrative Science Quarterly* (53:2), pp.333-362.

Zadeh, L. A. 1965. “Fuzzy Sets,” *Information and Control* (8:3), pp. 338-353.

Zammuto, R. F., Griffith, T. L., Majchrzak, A., Dougherty, D. J., and Faraj, S. 2007. “Information Technology and the Changing Fabric of Organization,” *Organization Science* (18:5), pp. 749–762.

Zhang, C., Hahn, J., and De, P. 2013. “Research Note—Continued Participation in Online Innovation Communities: Does Community Response Matter Equally for Everyone?” *Information Systems Research* (24:4), pp.1112-1130.

APPENDIX A. DESCRIPTIVE STATISTICS AND CORRELATION MATRIX FOR OTHER SAMPLES

Table A.1 Descriptive Statistics for Variables (Feature-Only Sample) – Sample 1

	Collective Attention Conditions				**Contingent Conditions**					**Productivity**	
	Attention partition by issues	Attention partition by modules	Attention augmentation by issues	Attention augmentation by modules	Number of issues	Number of modules	Number of developers	Project maturity	Length of release cycle	Speed	Novelty
Obs.	2134	2134	2134	2134	2134	2134	2134	2134	2134	2134	2.134
Mean	0.693	0.631	0.112	0.536	24.012	61.846	9.693	679.537	117.449	0.231	5094.747
Std.	0.219	0.237	0.173	0.296	53.557	87.137	17.932	656.107	226.118	0.363	37028.75
Min	0.000	0.000	0.000	0.000	1.000	1.000	2.000	0.000	14.000	0.000	0.000
25th	0.500	0.500	0.000	0.347	4.000	13.000	3.000	215.250	28.000	0.038	189.025
50th	0.741	0.667	0.000	0.593	9.000	31.000	4.000	506.000	51.000	0.105	562.975
75th	0.867	0.811	0.167	0.774	23.000	76.000	9.000	918.000	112.000	0.259	1696.655
Max	0.994	0.978	1.000	1.000	912.000	949.000	250.000	3780.000	3444.000	3.950	1256784

Notes: Descriptive statistics are based on uncalibrated measures. The number of observations for each variable is 2134 (see Obs.).

Table A.2 Correlation Matrix of Variables (Feature-Only Sample) – Sample 1

Variable	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1. Attention partition by issues	1.000										
2. Attention partition by modules	0.539	1.000									
3. Attention augmentation by issues	0.078	0.086	1.000								
4. Attention augmentation by modules	0.343	0.291	0.352	1.000							
5. Number of issues	0.347	0.314	0.099	0.250	1.000						
6. Number of modules	0.384	0.433	0.132	0.291	0.500	1.000					
7. Number of developers	0.347	0.387	0.069	0.208	0.826	0.546	1.000				
8. Project maturity	0.124	0.140	-0.076	-0.001	0.076	0.106	0.189	1.000			
9. Length of release cycle	0.128	0.059	0.050	0.059	0.165	0.122	0.111	-0.151	1.000		
10. Speed	0.344	0.338	0.073	0.285	0.420	0.418	0.439	0.084	-0.166	1.000	
11. Novelty	0.018	0.044	-0.025	-0.050	-0.008	0.172	0.005	-0.003	0.009	0.008	1.000

Notes: Correlations are based on uncalibrated measures.

Table A.3 Descriptive Statistics for Variables (Primary-Focus Redefined) – Sample 2

	Collective Attention Conditions				Contingent Conditions					Productivity	
	Attention partition by issues	Attention partition by modules	Attention augmentation by issues	Attention augmentation by modules	Number of issues	Number of modules	Number of developers	Project maturity	Length of release cycle	Speed	Novelty
Obs.	3052	3052	3052	3052	3052	3052	3052	3052	3052	3052	3052
Mean	0.824	0.783	0.075	0.434	61.111	70.409	15.243	674.943	96.802	0.769	5332.983
Std.	0.141	0.140	0.118	0.277	155.386	102.549	31.776	607.077	180.195	1.315	78506.53
Min	0.000	0.000	0.000	0.000	1.000	1.000	2.000	0.000	14.000	0.000	0.000
25th	0.750	0.722	0.000	0.226	8.000	15.000	3.000	231.500	25.000	0.107	127.164
50th	0.860	0.813	0.013	0.455	21.000	34.000	6.000	530.000	45.000	0.286	405.091
75th	0.929	0.883	0.109	0.650	55.250	81.250	13.000	935.000	98.000	0.763	1261.810
Max	0.996	0.987	0.889	1.000	3032.000	1285.000	439.000	3780.000	3444.000	14.377	4124507

Notes: Descriptive statistics are based on uncalibrated measures. The number of observations for each variable is 3052 (see Obs.).

Table A.4 Correlation Matrix of Variables (Primary Focus Redefined) – Sample 2

Variable	1	2	3	4	5	6	7	8	9	10	11
1. Attention partition by issues	1.000										
2. Attention partition by modules	0.583	1.000									
3. Attention augmentation by issues	0.215	0.183	1.000								
4. Attention augmentation by modules	0.346	0.381	0.463	1.000							
5. Number of issues	0.320	0.292	0.122	0.158	1.000						
6. Number of modules	0.365	0.447	0.196	0.243	0.510	1.000					
7. Number of developers	0.353	0.353	0.099	0.084	0.768	0.580	1.000				
8. Project maturity	0.093	0.096	-0.047	-0.028	0.155	0.164	0.238	1.000			
9. Length of release cycle	0.158	0.070	0.028	0.002	0.143	0.145	0.067	-0.133	1.000		
10. Speed	0.389	0.374	0.178	0.242	0.474	0.500	0.564	0.218	-0.141	1.000	
11. Novelty	-0.007	0.036	-0.010	0.016	-0.009	0.066	0.005	-0.008	-0.004	-0.007	1.000

Notes: Correlations are based on uncalibrated measures.

Table A.5 Descriptive Statistics for Variables (Peripheral Developers Excluded) – Sample 3

	Collective Attention Conditions				Contingent Conditions					Productivity	
	Attention partition by issues	Attention partition by modules	Attention augmentation by issues	Attention augmentation by modules	Number of issues	Number of modules	Number of developers	Project maturity	Length of release cycle	Speed	Novelty
Obs.	2813	2813	2813	2813	2813	2813	2813	2813	2813	2813	2813
Mean	0.738	0.645	0.151	0.669	62.049	75.081	12.453	673.647	100.175	0.772	5607.760
Std.	0.200	0.231	0.182	0.262	155.009	105.639	24.256	616.903	186.300	1.286	81966.97
Min	0.000	0.000	0.000	0.000	1.000	1.000	2.000	0.000	14.000	0.000	0.000
25th	0.656	0.500	0.000	0.550	9.000	18.000	3.000	229.000	26.000	0.107	159.250
50th	0.781	0.667	0.083	0.748	22.000	37.000	5.000	518.000	47.000	0.302	459.500
75th	0.890	0.816	0.250	0.861	57.000	88.000	11.000	928.000	101.000	0.800	1411.614
Max	0.995	0.982	1.000	1.000	2938.000	1285.000	326.000	3780.000	3444.000	14.377	4124507

Notes: Descriptive statistics are based on uncalibrated measures. The number of observations for each variable is 2813 (see Obs.).

Table A.6 Correlation Matrix of Variables (Peripheral Developers Excluded) – Sample 3

Variable	1	2	3	4	5	6	7	8	9	10	11
1. Attention partition by issues	1.000										
2. Attention partition by modules	0.578	1.000									
3. Attention augmentation by issues	0.142	0.134	1.000								
4. Attention augmentation by modules	0.371	0.285	0.331	1.000							
5. Number of issues	0.328	0.287	0.109	0.207	1.000						
6. Number of modules	0.382	0.413	0.185	0.239	0.496	1.000					
7. Number of developers	0.392	0.376	0.156	0.205	0.760	0.592	1.000				
8. Project maturity	0.153	0.151	-0.002	0.043	0.156	0.172	0.255	1.000			
9. Length of release cycle	0.135	0.057	0.015	0.041	0.137	0.136	0.045	-0.135	1.000		
10. Speed	0.389	0.366	0.145	0.279	0.469	0.482	0.574	0.221	-0.149	1.000	
11. Novelty	0.005	0.035	0.021	0.003	-0.010	0.060	0.005	-0.011	-0.0059	-0.009	1.000

Notes: Correlations are based on uncalibrated measures.